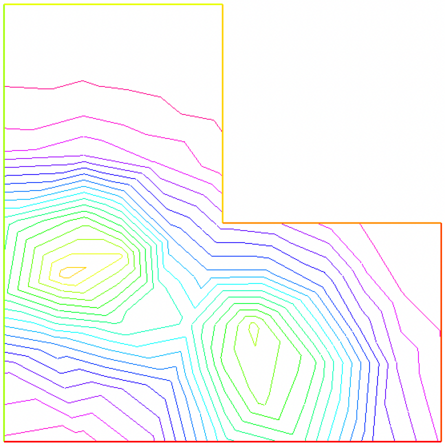
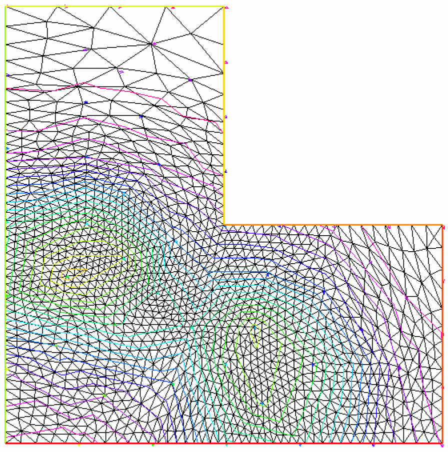
Faculté des sciences de Montpellier
Rapport séance 5

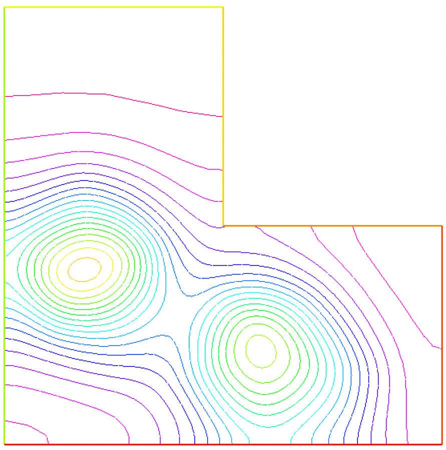
Étudiant: Giscard Leonel Zouakeu

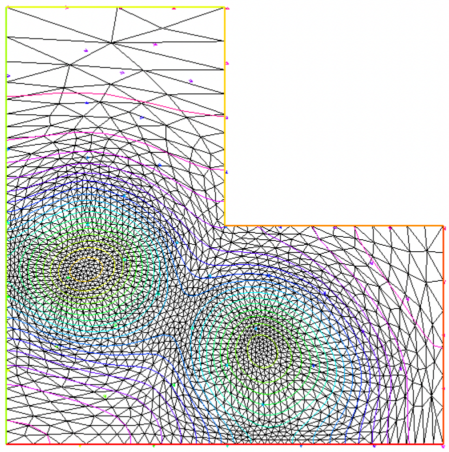
1-COMPRÉHENSION ET EXPLICATION DU CODE: "*adapsteady.edp*"

```
border a(t=0,1.0){x=t;   y=0;  label=1;};// comment
border b(t=0,0.5){x=1;   y=t;  label=2;};
border c(t=0,0.5){x=1-t; y=0.5;label=3;};
border d(t=0.5,1){x=0.5; y=t;  label=4;};
border e(t=0.5,1){x=1-t; y=1;  label=5;};
border f(t=0.0,1){x=0;   y=1-t;label=6;};
mesh Th = buildmesh (a(6) + b(4) + c(4) +d(4) + e(4) + f(6));
//savemesh(Th,"th.msh");
fespace Vh(Th,P1);




Vh u,v,source;
Vh vit1 = y*0.01, vit2 = -x*0.01;              // velocity
real error=0.005,xnu=0.01,xlam=1;
source=exp(-100*((x-0.2)^2+(y-0.4)^2))+exp(-100*((x-0.6)^2+(y-0.2)^2));

problem Probem1(u,v,solver=CG,eps=1.0e-6) =
    int2d(Th,qforder=2)( u*v*1.0e-10+ xnu*( dx(u)*dx(v) + dy(u)*dy(v) ))
  + int2d(Th,qforder=2)( (-u*vit1*dx(v) - u*dy(v)*vit2 ))
  + int2d(Th,qforder=2)(  source*v)
  + int2d(Th,qforder=2)(  xlam*u*v);

for (int iadap=0;iadap< 10;iadap++)
{
  Probem1;
   cout << u[].min << " " << u[].max << endl;
   plot(u,wait=1);
   Th=adaptmesh(Th,u,err=error,hmin=0.01,hmax=0.5);
   plot(Th,u,[vit1,vit2],wait=1);
  u=u;                 // interpolation from old to new meshes
 //error = error/2;
} ;
```









D'après notre code adapsteady.edp , la formulation variationnelle du problème s'écrit:

$$\left(\int_\Omega u.v + \nu \int_\Omega \nabla u \nabla v - \int_\Omega u.V(t,x)\nabla v + \int_\Omega f.v + \lambda \int_\Omega u.v\right) = 0 \qquad (1)$$

avec v appartenant à l'espace des fonctions test, plus precisement $v \in \mathbf{H}_0^1(\Omega)$. Nous avons la fonction source "f" définie dans notre code par:

$$f(x,y) = \exp(-100*((x-0.2)^2+(y-0.4)^2))+\exp(-100*((x-0.6)^2+(y-0.2)^2))$$

et donc le problème lié a cette formulation variationnelle (1) s'écrit:

$$-\nu\Delta u - \lambda u + V(t,x)\nabla u = f \text{ sur } \Omega \tag{2}$$
$$u = 0 \text{ sur } \partial\Omega$$

2- PROBLÈME DE STOKES

Soit $\Omega \in \mathbf{R^2}$ et $\mathbf{f}$,$\mathbf{U}_\mathrm{D}$ deux fonctions de $\mathbf{R^2} \to \mathbf{R^2}$. On considère le problème: Chercher deux fonctions $\mathbf{u} : \mathbf{R^2} \to \mathbf{R^2}$ et $p : \mathbf{R^2} \to \mathbf{R^2}$ telles que :

$$\begin{aligned} -\Delta \mathbf{U} + \nabla p &= \mathbf{f} \quad \text{dans } \Omega \\ \nabla \mathbf{U} &= 0 \quad \text{dans } \Omega \\ \mathbf{U} &= \mathbf{U}_\mathrm{D} \quad \text{sur } \partial\Omega \end{aligned} \tag{3}$$

Ou , pour $\mathbf{U} = (u_1, u_2)^t$, on a posé $\mathbf{\Delta U} = (\Delta u_1, \Delta u_2)^t$,$\nabla\mathbf{U} = (\frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial x})$. Et pour toute fonction $v : \mathbf{R^2} \to \mathbf{R}$, $\Delta v = \frac{\partial^2 u_1}{\partial x^2} + \frac{\partial^2 u_2}{\partial x^2}$.

Formulation variationnelle du problème de Stokes

On pose:
$\mathbf{H^1}(\Omega) = \{v \in \mathbf{L^2}(\Omega), \nabla v \in (\mathbf{L^2}(\Omega))^2\}; \mathbf{H_0^1}(\Omega) = \{v \in \mathbf{H^1}(\Omega); v = 0 \text{ sur } \partial\Omega\}$

$$V = \mathbf{H^1}(\Omega) \times \mathbf{H_0^1}(\Omega); M = \{q \in \mathbf{L^2}(\Omega); \int_\Omega q.\mathbf{d}x\mathbf{d}y = 0\}$$

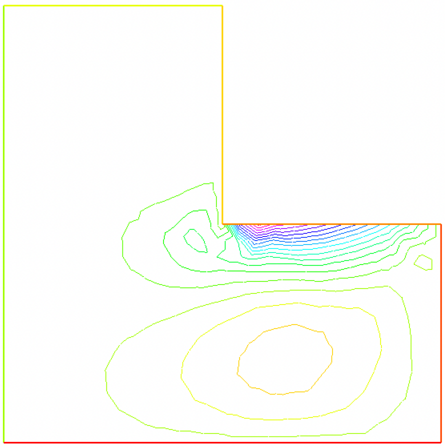
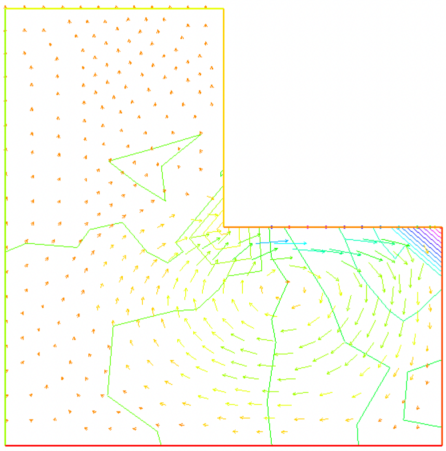
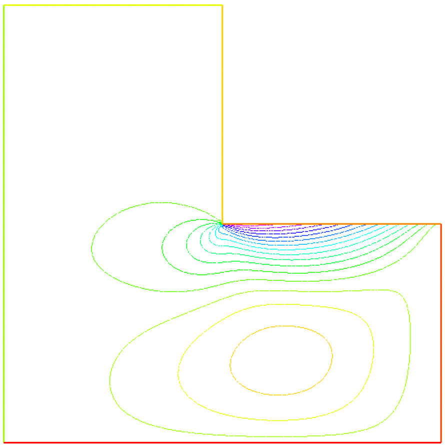
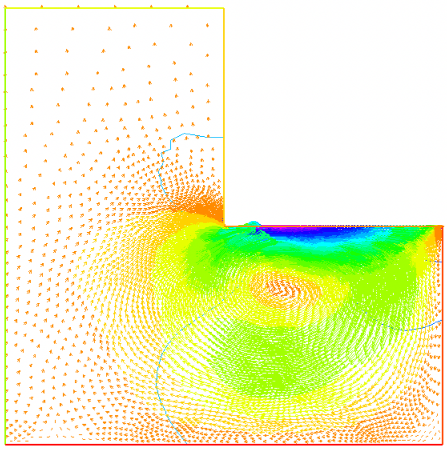
$$a(\mathbf{u}, \mathbf{v}) = \int_\Omega \nabla\mathbf{u}\nabla\mathbf{v}\mathbf{d}x\mathbf{d}y; \quad b(\mathbf{u}, p) = -\int_\Omega p\nabla.\mathbf{v}\mathbf{d}x\mathbf{d}x$$

$$l(\mathbf{v}) = \int_\Omega \mathbf{f}.\mathbf{v}\mathbf{d}x\mathbf{d}y$$

Code adapsteady.edp adapté au problème de Stokes.

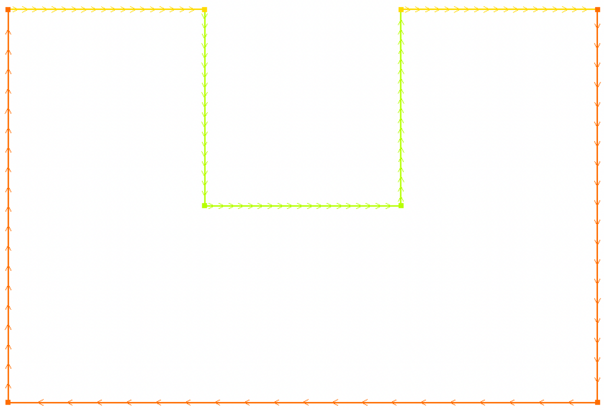
```
border a(t=0,1.0){x=t;   y=0;  label=1;};// comment
border b(t=0,0.5){x=1;   y=t;  label=2;};
border c(t=0,0.5){x=1-t; y=0.5;label=3;};
border d(t=0.5,1){x=0.5; y=t;  label=4;};
border e(t=0.5,1){x=1-t; y=1;  label=5;};
border f(t=0.0,1){x=0;   y=1-t;label=6;};
mesh Th = buildmesh (a(6) + b(4) + c(4) +d(4) + e(4) + f
   (6));

//Expace d'approximation de chaque composante de la
   vitess
fespace Vh(Th,P2);
//Expace d'approximation de la pression
fespace Wh(Th,P1);
Vh u1,u2; //composantes de la vitesse
```

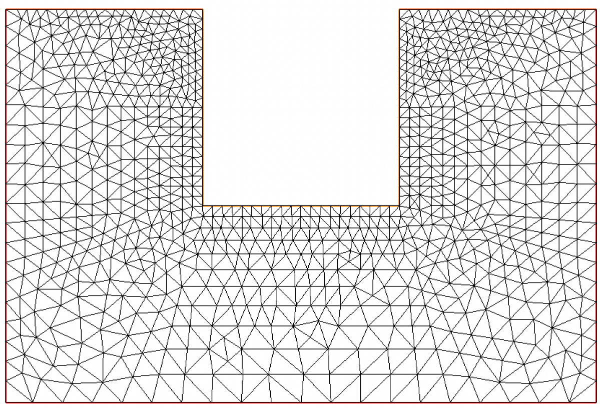
```
Vh v1,v2; //fonctions tests
Wh p; //pressiona‘calculer
Wh q; //fonction test


real nu=1;

func g=(x)*(1-x)*4;
func f1 =exp(-100*((x-0.2)^2+(y-0.4)^2))+exp(-100*((x
    -0.6)^2+(y-0.2)^2));
func f2 =exp(-100*((x-0.2)^2+(y-0.4)^2))+exp(-100*((x
    -0.6)^2+(y-0.2)^2));
real error=0.005,xnu=0.01,xlam=1;
//e‘Problme variationnel
problem Stokes ([u1,u2,p],[v1,v2,q],solver=Crout) =
int2d(Th)(
nu * ( dx(u1)*dx(v1) + dy(u1)*dy(v1)
+ dx(u2)*dx(v2) + dy(u2)*dy(v2) )
+ p*q*(0.000001) /* ne pas oublier ce terme */ - p*dx(v1)
     - p*dy(v2)
- dx(u1)*q - dy(u2)*q
)
+ int2d(Th) ( -f1*v1 - f2*v2 )
+ on(3,u1=g,u2=0)
+ on(1,2,4,u1=0,u2=0);
//Resolution
//Stokes;
//postprocessing
//plot(coef=0.2,cmm=" [u1,u2] et p ",p,[u1,u2],ps="
    StokesP2P1.eps",value=1,wait=1);

for (int iadap=0;iadap< 10;iadap++)
{
  Stokes;
   cout << u1[].min << " " << u1[].max << endl;
   plot(u1,wait=1);
   Th=adaptmesh(Th,u1,err=error,hmin=0.01,hmax=0.5);
   plot(coef=0.2,cmm=" [u1,u2] et p ",p,[u1,u2],ps="
       StokesP2P1.eps",value=1,wait=1);
  u1=u1;                // interpolation from old to new
      meshes
 //error = error/2;
} ;
```

2- PROBLÈME DE STOKES SOUS FORME DE U

Domaine et maillage du domaine sous forme de U.

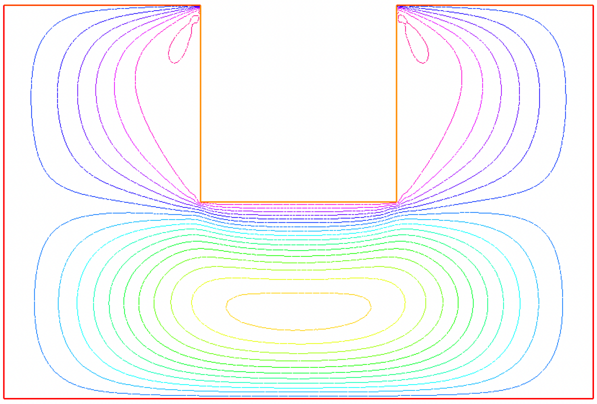


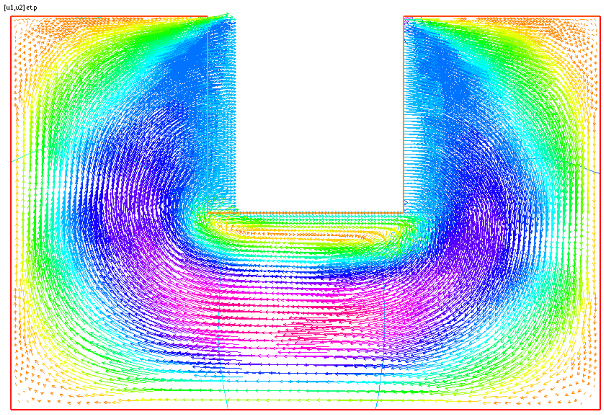


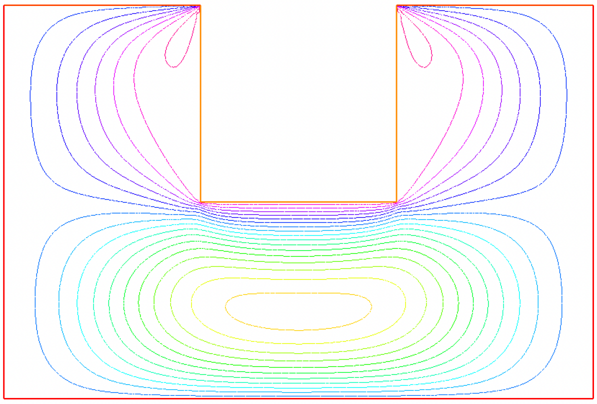
Code équation Stokes avec domaine sous forme de U

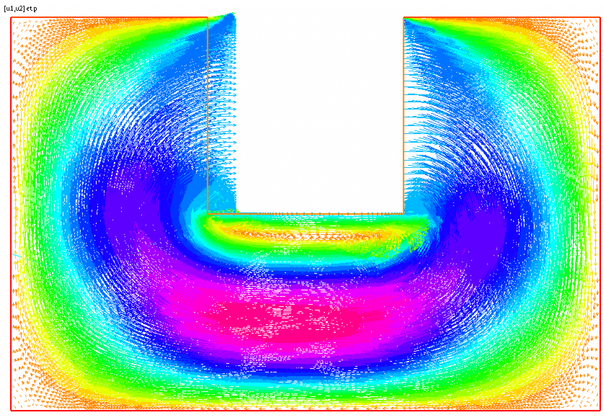
```
int upper = 1;
int others = 2;
int inner = 3;

border C01(t=0, 1){x=0; y=-1+t; label=upper;}
border C02(t=0, 1){x=1.5-1.5*t; y=-1; label=upper;}
border C03(t=0, 1){x=1.5; y=-t; label=upper;}
border C04(t=0, 1){x=1+0.5*t; y=0; label=others;}
border C05(t=0, 1){x=0.5+0.5*t; y=0; label=others;}
border C06(t=0, 1){x=0.5*t; y=0; label=others;}
border C11(t=0, 1){x=0.5; y=-0.5*t; label=inner;}
border C12(t=0, 1){x=0.5+0.5*t; y=-0.5; label=inner;}
border C13(t=0, 1){x=1; y=-0.5+0.5*t; label=inner;}

int n = 20;
plot(C01(-n) + C02(-n) + C03(-n) + C04(-n)
     + C06(-n) + C11(n) + C12(n) + C13(n), wait=true);

mesh Th = buildmesh(C01(-n) + C02(-n) + C03(-n) + C04(-n)
 + C06(-n) + C11(-n) + C12(-n) + C13(-n));

plot(Th, wait=true);


//// Stokes ////
fespace Vh(Th, P2); //definition of the velocity
    component space
fespace Wh(Th, P1); //definition of the pressure space
Vh u2, v2;
Vh u1, v1;
Wh p, q;
func g= 1;
```

```
func f1 =exp(-100*((x-0.2)^2+(y-0.4)^2))+exp(-100*((x-0.6)^2+(y-0.2)^2));
func f2 =exp(-100*((x-0.2)^2+(y-0.4)^2))+exp(-100*((x-0.6)^2+(y-0.2)^2));
real error=0.005,xnu=0.01,xlam=1;

solve Stokes ([u1,u2,p],[v1,v2,q],solver=Crout)
= int2d(Th)(
 (
 dx(u1)*dx(v1)
 + dy(u1)*dy(v1)
 + dx(u2)*dx(v2)
 + dy(u2)*dy(v2)
 )
 - p*q*(0.000001)
 - p*dx(v1) - p*dy(v2)
 - dx(u1)*q - dy(u2)*q
 )
 + int2d(Th) ( -f1*v1 - f2*v2 )

+ on(3,u1=g,u2=0)
+ on(1,2,4,u1=0,u2=0);

for (int iadap=0;iadap< 10;iadap++)
{
  Stokes;
   cout << " " << u1[].min << " " << u1[].max << endl;
   plot(u1,wait=1);
   Th=adaptmesh(Th,u1,err=error,hmin=0.01,hmax=0.5);
   plot(coef=0.2,cmm=" [u1,u2] et p ",p,[u1,u2],ps="StokesP2P1.eps",wait=1);
  u1=u1;                // interpolation from old to new meshes
 //error = error/2;
} ;
```

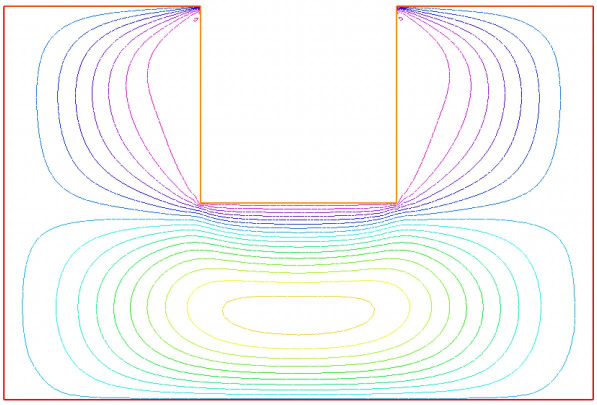


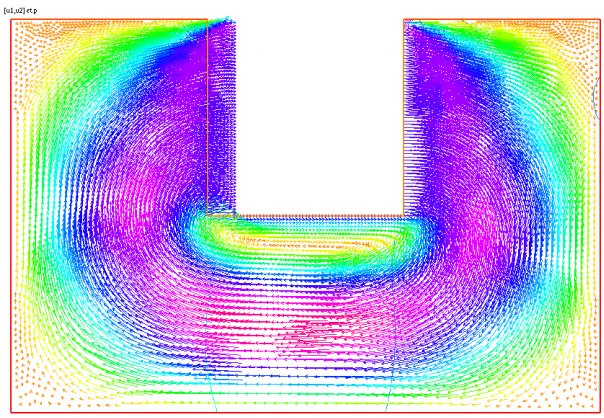


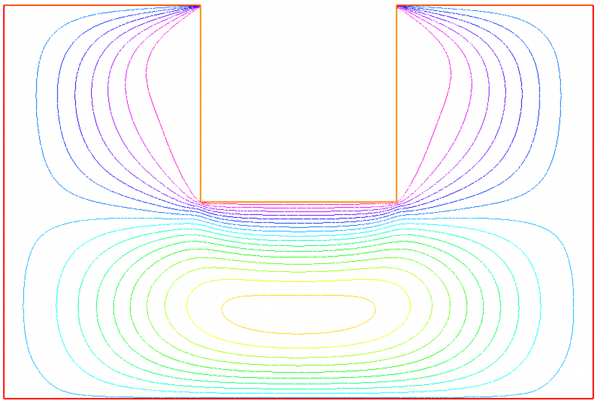
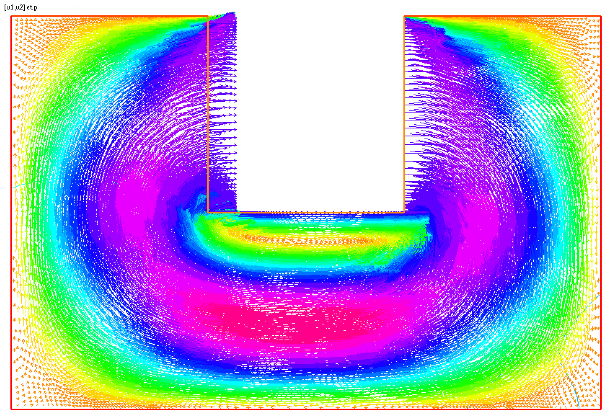




Probleme Stokes pour un niveau d'erreur de epsilon = 0,01 et 0.005





2- PROBLÈME ADRS SOUS FORME DE U

```
int upper = 1;
int others = 2;
int inner = 3;

border C01(t=0, 1){x=0; y=-1+t; label=upper;}
border C02(t=0, 1){x=1.5-1.5*t; y=-1; label=upper;}
border C03(t=0, 1){x=1.5; y=-t; label=upper;}
border C04(t=0, 1){x=1+0.5*t; y=0; label=others;}
border C05(t=0, 1){x=0.5+0.5*t; y=0; label=others;}
border C06(t=0, 1){x=0.5*t; y=0; label=others;}
border C11(t=0, 1){x=0.5; y=-0.5*t; label=inner;}
border C12(t=0, 1){x=0.5+0.5*t; y=-0.5; label=inner;}
border C13(t=0, 1){x=1; y=-0.5+0.5*t; label=inner;}

int n = 20;
plot(C01(-n) + C02(-n) + C03(-n) + C04(-n)
     + C06(-n) + C11(n) + C12(n) + C13(n), wait=true);

mesh Th = buildmesh(C01(-n) + C02(-n) + C03(-n) + C04(-n)
 + C06(-n) + C11(-n) + C12(-n) + C13(-n));

plot(Th, wait=true);
//savemesh(Th,"th.msh");
fespace Vh(Th,P1);




Vh u,v,source;
```

```
Vh vit1 = y*0.01, vit2 = -x*0.01;                    // velocity
real error=0.005,xnu=0.01,xlam=1;
source=exp(-100*((x-0.2)^2+(y-0.4)^2))+exp(-100*((x-0.6)^2+(y-0.2)^2));

problem Probem1(u,v,solver=CG,eps=0.01) =
    int2d(Th,qforder=2)( u*v*1.0e-10+ xnu*( dx(u)*dx(v) + dy(u)*dy(v) ))
  + int2d(Th,qforder=2)( (-u*vit1*dx(v) - u*dy(v)*vit2 ))
  + int2d(Th,qforder=2)(  source*v)
  + int2d(Th,qforder=2)(  xlam*u*v);

for (int iadap=0;iadap< 10;iadap++)
{
  Probem1;
   cout << u[].min << " " << u[].max << endl;
   plot(u,wait=1);
   Th=adaptmesh(Th,u,err=error,hmin=0.01,hmax=0.5);
   plot(Th,u,[vit1,vit2],wait=1);
  u=u;               // interpolation from old to new meshes
 //error = error/2;
} ;
```