FACULTÉ DES SCIENCES
MONTPELLIER

Faculté des sciences de Montpellier
Rapport séance 6

Étudiant: Giscard Leonel Zouakeu

Problème inverse (cf poly.pdf bijan Mohammadi)

On souhaite analyser l'impact de la précision du calcul de la solution du problème direct sur la solution d'un problème inverse faisant intervenir ce problème direct.

Problème

On considère le problème le problème (définie dans notre) suivant:

$$-\Delta \mathbf{u} - \lambda \mathbf{u} = \mathbf{f} \quad \text{sur } \Omega$$
$$\mathbf{u} = \sin(x.y) \quad \text{sur } \partial\Omega$$

- Le terme source $\mathbf{f}$ contient N sources de la forme:

$$\alpha_i \delta(s_i) \quad (\sim \alpha_i \exp(-\beta(s - s_i)^2), \beta >> 0)$$

que l'on souhaite optimiser:

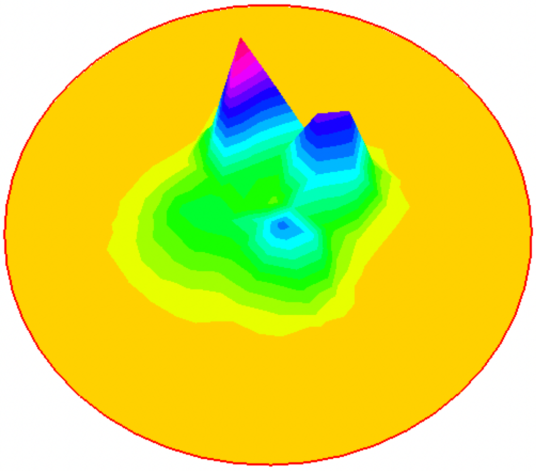
$$\mathbf{f} = \sum_{i=0}^{N} \alpha_i \delta(s_i)$$

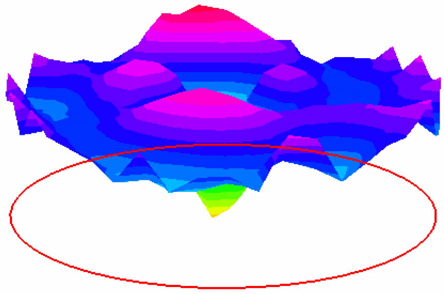
•En utilisant le principe de superposition, l'on peut décomposer la solution inconnue en la combinaison de $N + 1$ solutions:

$$\mathbf{u}_h = \mathbf{u}_{0,h} + \sum_{i=0}^{N} \alpha_i \mathbf{u}_{i,h}$$

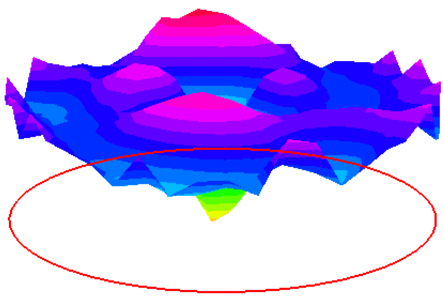
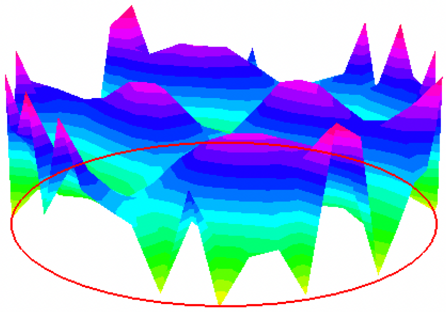
```
///
/// inverse problem example
///
// Parametres source terms
int nbctrl=4;  // later to make free the nbre of ctrl
real[int] alpha(nbctrl);

real alpha1=1;
real alpha2=2;
real alpha3=3;
real alpha4=4;
real beta=1;
real xx1=-6;
real xx2=-4;
real xx3=4;
real xx4=6;
real dt=0.1;
int iwait;
```

```
// PDE Parameters:

real lambda=1; //reaction coef.

// Boundary Conditions

func bc=sin(x*y);
int ibc=1;

border Omega1(t=0,2*pi){x=10*cos(t);y=10*sin(t);}
mesh Th1=buildmesh(Omega1(50));

fespace Vh1(Th1,P1);

/////////////////////////////////////////////
/// PB DEF :  ADV - DIFF - REACT
//////////////////////////////////////////

Vh1 u,v,ff;
problem ARD(u,v)=
int2d(Th1)(dx(u)*dx(v) + dy(u)*dy(v)) //Diffusion
//+int2d(Th1)(-y*dx(v)+x*dy(v)) //Convection  with [-y,x]
-int2d(Th1)(lambda*u*v) // Reaction
-int2d(Th1)(ff*v) //  source
+on(Omega1,u=bc*ibc); //BC

////////////// TARGET ///////////////
////// -L u= f1+f2+f3+f4

Vh1 udes,
f1=exp(-beta*0.2*((x-2.5)^2+(y)^2)),
f2=exp(-beta*0.5*((x)^2+(y-2.5)^2)),
f3=exp(-beta*0.5*((x+2.5)^2+(y)^2)),
f4=exp(-beta*0.8*((x)^2+(y+2.5)^2));
ibc=1;
ff=alpha1*f1+alpha2*f2+alpha3*f3+alpha4*f4-convect([-y,x
    ],-dt,u);
ARD;

plot(Th1,wait=iwait);
plot(ff,dim=3,fill=1,wait=iwait);
plot(u,dim=3,fill=1,wait=iwait);
```





```
Vh1 uL;
u=uL;
ff=-convect([-y,x],-dt,uL);
ibc=1;
ARD;
uL=u;

plot(uL,dim=3,fill=1,wait=iwait);

/////////////// Solution U1

Vh1 u1;
ff=-convect([-y,x],-dt,u1)+f1;
u=u1;
ibc=0;
ARD;
u1=u;
plot(u1,dim=3,fill=1,wait=iwait);
```

```
/////////////// Solution U2

Vh1 u2;
ff=-convect([-y,x],-dt,u2)+f2;
u=u2;
ibc=0;
ARD;
u2=u;
plot(u2,dim=3,fill=1,wait=iwait);

/////////////// Solution U3 (f1terme source)

Vh1 u3;
ff=-convect([-y,x],-dt,u3)+f3;
u=u3;
ibc=0;
ARD;
u3=u;
plot(u3,dim=3,fill=1,wait=iwait);


/////////////// Solution U4

Vh1 u4;
ff=-convect([-y,x],-dt,u4)+f4;
u=u4;
ibc=0;
ARD;
u4=u;
plot(u4,dim=3,fill=1,wait=iwait);
```

Calcul numerique de l'expression A_{ij}, B_i

En écrivant les conditions d'optimalité d'ordre un de notre problème de moindres carrés, on obtient les coefficients $\alpha_i = 1, ..., N$ en résolvant un système linéaire de N équations à N inconnues: $\mathrm{Ax} = \mathrm{B}$ avec:

$$A_{i,j} = \int_{\omega_h} \mathbf{u}_{i,h}(x,s)\mathbf{u}_{j,h}(x,s)\mathbf{d}s$$

$$B_i = \int_{\omega_h} (\mathbf{u}_{0,h}(s) - \mathbf{u}_{des,h}(s))\mathbf{u}_{j,h}(x,s)\mathbf{d}s \quad i,j = 1 \ldots\ldots N$$

```
///////////  MATRICE A & F ///////////
/// Aij= <ui,uj>L2

real a11=int2d(Th1)(u1*u1);
real a12=int2d(Th1)(u1*u2);
real a13=int2d(Th1)(u1*u3);
real a14=int2d(Th1)(u1*u4);
real a22=int2d(Th1)(u2*u2);
real a23=int2d(Th1)(u2*u3);
real a24=int2d(Th1)(u2*u4);
real a33=int2d(Th1)(u3*u3);
real a34=int2d(Th1)(u3*u4);
real a44=int2d(Th1)(u4*u4);


real F1=int2d(Th1)(u1*(uL-udes));
real F2=int2d(Th1)(u2*(uL-udes));
real F3=int2d(Th1)(u3*(uL-udes));
real F4=int2d(Th1)(u4*(uL-udes));


cout << "a11 = " << a11 << endl;
cout << "a12 = " << a12 << endl;
cout << "a13 = " << a13 << endl;
cout << "a14 = " << a14 << endl;
cout << "a22 = " << a22 << endl;
cout << "a23 = " << a23 << endl;
cout << "a24 = " << a24 << endl;
cout << "a33 = " << a33 << endl;
cout << "a34 = " << a34 << endl;
cout << "a44 = " << a44 << endl;

cout << "F1 = " << F1 << endl;
cout << "F2 = " << F2 << endl;
cout << "F3 = " << F3 << endl;
```

```
cout << "F4 = " << F4 << endl;


real[int,int] A(4,4);
A=[[a11,a12,a13,a14],[a12,a22,a23,a24],[a13,a23,a33,a34
    ],[a14,a24,a34,a44]];
matrix B=A;
real[int] b=[-F1,-F2,-F3,-F4];
real[int] c(b.n);
c=B*b;
set(B,solver=UMFPACK);
real[int] d(4);
d=B^-1*b;

cout <<"solution" << d<< endl;
```

Solution finale du problème inverse

```
Vh1 uopt;
u=uopt;
ff=d(0)*f1+d(1)*f2+d(2)*f3+d(3)*f4-convect([-y,x],-dt,
    uopt);
ibc=1;
ARD;
uopt=u;

real J0=int2d(Th1)((uL-udes)^2);
real J=int2d(Th1)((uopt-udes)^2);

Vh1 err=udes-uopt;
//plot(uopt,dim=3,fill=1,wait=1);
plot(err,dim=3,fill=1,wait=1);


cout <<"cost init / end     " << J0<< "   ,   "<< J << endl
```