EE-206 Exercice 2 - Plan de Doehlert

Table of Contents

Enoncé

Avec l'objectif d'optimiser un dépôt d'une couche mince sur un wafer de silicium, des expériences selon un plan de Doehlert ont été réalisées. L'expérience consiste à placer un wafer dans un four maintenu dans des conditions déterminées de température et de pression et d'injecter différents gaz en relation avec la nature du dépôt souhaité. Dans la situation présente, trois facteurs ont été variés, à savoir la concentration de C_2H_2, la concentration de CO_2 et la pression dans le four. Le résultat mesuré est l'énergie de surface qui est en lien avec la qualité du dépôt. Les données de l'expérience sont disponibles dans la table 1.

Questions

1. ...
2. ...
3. Construire la matrice standardisée et non-standardisée du modèle quadratique
4. Inférer les effets (standard et non-standard) pour les résultats expérimentaux donnés en annexe
5. Réaliser une analyse canonique du modèle non-standard et déterminer la forme canonique du modèle.
6. Réaliser un graphe 3D slice du modèle standard avec 3 plans passant par le point fixe.
7. Réaliser un graphique 3D d'une surface d'isoréponse pour illustrer la géométrie du modèle.

Point 3

Plan d'expériences

```
E_std=doehlert(3);
disp ('Plan d''expériences standardisé')
```

```
Plan d'expériences standardisé
```

```
disp(E_std)
```

```
         0         0         0
   -1.0000         0         0
   -0.5000   -0.8660         0
   -0.5000   -0.2887   -0.8165
    1.0000         0         0
    0.5000    0.8660         0
    0.5000    0.2887    0.8165
   -0.5000    0.8660         0
   -0.5000    0.2887    0.8165
         0   -0.5774    0.8165
    0.5000   -0.8660         0
    0.5000   -0.2887   -0.8165
         0    0.5774   -0.8165
```

Plages de variations des facteurs

```
RangeMin=[10 10 100];
RangeMAX=[30 16 200];
```

Plan non-standardisé

```
E=rescale(E_std,[10 10 100],[30 16 200])
```

```
E = 13×3
   20.0000   13.0000  150.0000
   10.0000   13.0000  150.0000
   15.0000   10.4019  150.0000
   15.0000   12.1340  109.1752
   30.0000   13.0000  150.0000
   25.0000   15.5981  150.0000
   25.0000   13.8660  190.8248
   15.0000   15.5981  150.0000
   15.0000   13.8660  190.8248
   20.0000   11.2679  190.8248
      ⋮
```

```
disp ('Plan d''expériences non-standardisé')
```

```
Plan d'expériences non-standardisé
```

```
disp(E)
```

```
   20.0000   13.0000  150.0000
   10.0000   13.0000  150.0000
   15.0000   10.4019  150.0000
   15.0000   12.1340  109.1752
   30.0000   13.0000  150.0000
   25.0000   15.5981  150.0000
   25.0000   13.8660  190.8248
```

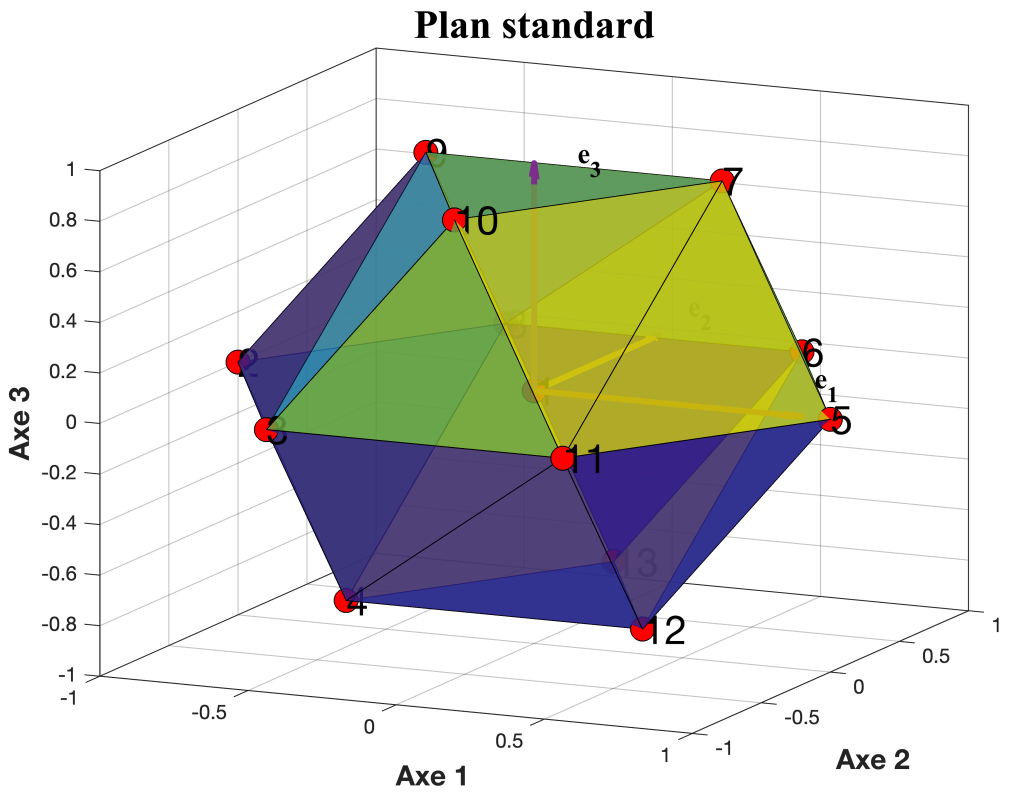
```
   15.0000   15.5981  150.0000
   15.0000   13.8660  190.8248
   20.0000   11.2679  190.8248
   25.0000   10.4019  150.0000
   25.0000   12.1340  109.1752
   20.0000   14.7321  109.1752
```

Axiométrie

Invoque la fonction de dessin d'un polyédre.

Cette fonction a un défaut mineur, de représenter les faces carrées par deux triangles; cela vient de l'utilisation de la fonction convhull() qui évite de devoir définir chacune des faces individuellement.

```
%
% Paramètres pour le dessin
azimut=25;
elevation=15;
titre='Plan standard';

S=dessin_3D(E_std,azimut,elevation,titre);
```



Point 4

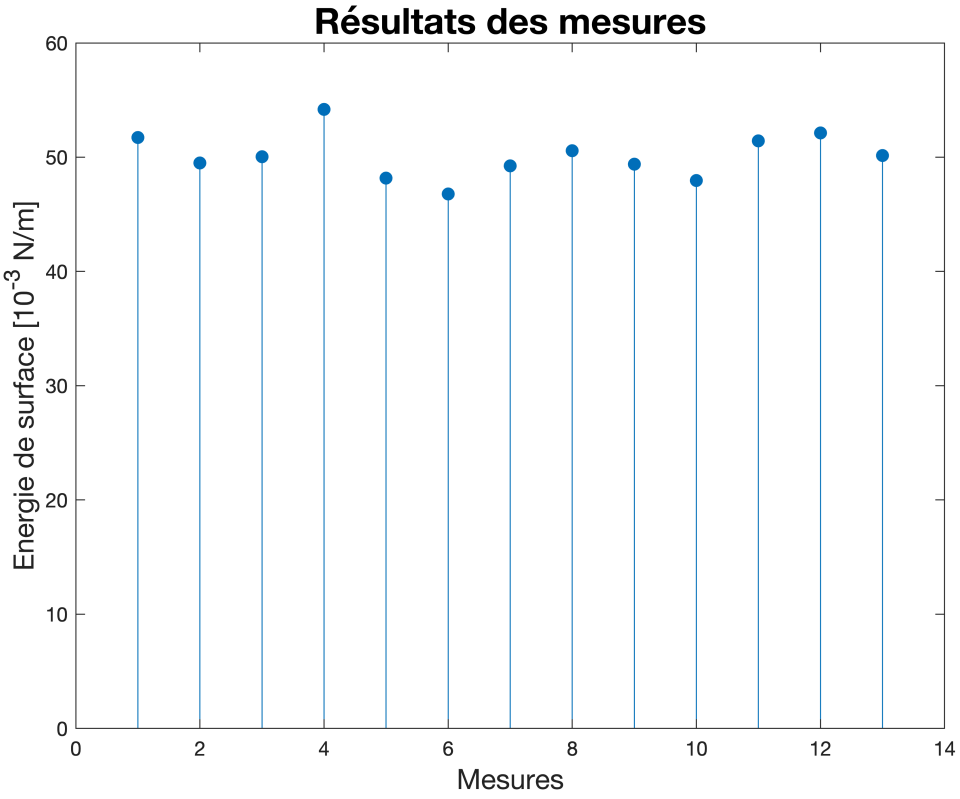
Résultats d'expérience

```
% Y=[51.739
% 51.464
% 51.220
```

```
% 51.720
% 46.072
% 45.891
% 51.719
% 52.331
% 45.826
% 48.831
% 49.899
% 55.758
% 49.274]

% série corrigée
Y=[51.72
   49.50
   50.05
   54.19
   48.17
   46.78
   49.25
   50.56
   49.38
   47.95
   51.42
   52.12
   50.14]
```

```
Y = 13×1
   51.7200
   49.5000
   50.0500
   54.1900
   48.1700
   46.7800
   49.2500
   50.5600
   49.3800
   47.9500
      ⋮
```

```
% Barchart des résultats
stem(Y,'filled')
title('Résultats des mesures','FontSize',18)
xlabel('Mesures','FontSize',14)
ylabel('Energie de surface [10^{-3} N/m]','FontSize',14)
```



Inférence des effets

On utilise la fonction *fitlm(x,y,modelspec)* pour effectuer la régression linéaire

```
mdl=fitlm(E,Y,"quadratic")
```

```
mdl =
Linear regression model:
    y ~ 1 + x1*x2 + x1*x3 + x2*x3 + x1^2 + x2^2 + x3^2

Estimated Coefficients:
                    Estimate         SE        tStat      pValue
                   __________    _________    _______    _________

    (Intercept)        47.717       12.345     3.8652     0.030626
    x1                 1.6798       0.2228     7.5394    0.0048375
    x2                 3.0917        1.242     2.4893     0.088537
    x3                -0.35342     0.058349     -6.057     0.009029
    x1:x2             -0.099112     0.011889    -8.3365     0.003618
    x1:x3             0.0044785    0.00079753     5.6154     0.011164
    x2:x3              0.023374     0.0026584     8.7923    0.0030995
    x1^2               -0.02885      0.003783    -7.6262    0.0046806
    x2^2               -0.19204      0.042034    -4.5687     0.019672
    x3^2            -0.00026767    0.00014706    -1.8202       0.1663


Number of observations: 13, Error degrees of freedom: 3
Root Mean Squared Error: 0.309
R-squared: 0.994,  Adjusted R-Squared: 0.975
```

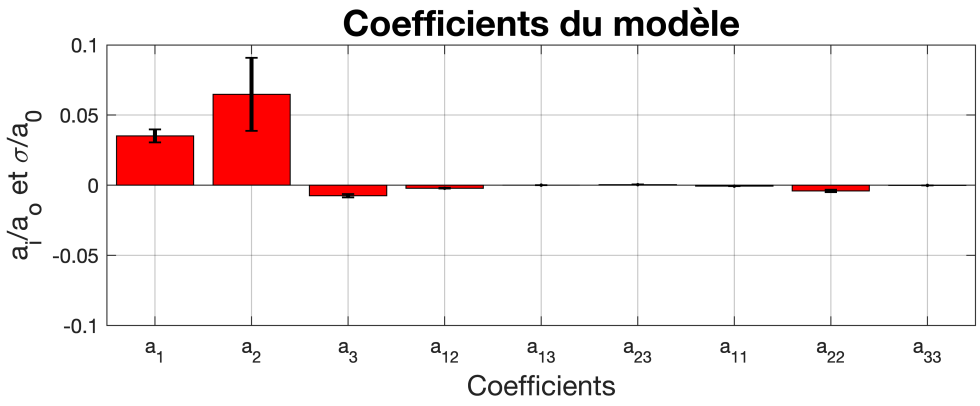
```
F-statistic vs. constant model: 53.7, p-value = 0.00372
```

On récupère les coefficients du modèle

```
coef=mdl.Coefficients.Estimate;
```

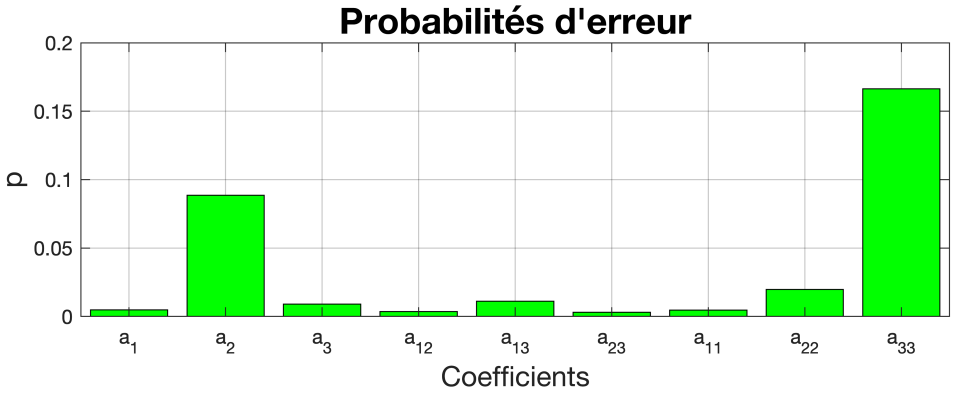
Diagramme en colonne des coefficients relatifs

```
subplot(211)
bar(coef(2:end)/coef(1),'red')
title('Coefficients du modèle','FontSize',18)
xlabel('Coefficients','FontSize',14)
ylabel('a_i/a_o et \sigma/a_0','FontSize',14)
xticklabels({'a_1','a_2','a_3','a_{12}','a_{13}','a_{23}','a_{11}','a_{22}','a_{33}'})
grid
axis([.5 9.5 -0.1 0.1]) % gestion de axes

% Ajout des barres d'erreur
hold on
errorbar(1:9,coef(2:end)/coef(1),...
    mdl.Coefficients.SE(2:end)/coef(1),...
    '.k','LineWidth',2)
hold off
```




```
figure
subplot(212)
bar(mdl.Coefficients.pValue(2:end),'green')
title('Probabilités d''erreur','FontSize',18)
```

```
xlabel('Coefficients','FontSize',14)
ylabel('p','FontSize',14)
xticklabels({'a_1','a_2','a_3',...
    'a_{12}','a_{13}','a_{23}',...
    'a_{11}','a_{22}','a_{33}'})
grid
axis([.5 9.5 0 0.2]) % gestion de axes
```



Ces résultats sont satisfaisants au niveau statistique à part pour les coefficients a_2 et a_{33}, cas pour lesquels la probabilité est supérieure aux 5% qui servent de critère générique.

```
```

Analyse canonique selon les axes standardisés

Constante a_o

```
ao=coef(1);
```

Effets linéaires $\vec{a}$

```
a=coef(2:4);
```

Matrice du second degré $\mathbf{A}$

```
A=[coef(8) coef(5)/2  coef(6)/2
```

```
    coef(5)/2 coef(9) coef(7)/2
    coef(6)/2 coef(7)/2 coef(10)];
```

Détermine les coordonnées du point fixe $X_s = -\frac{1}{2} A^{-1} \vec{a}$

```
Xs =-.5*A\a;
disp('Le point fixe')
```

```
Le point fixe
```

```
disp(Xs)
```

```
   68.0334
   67.2417
  864.2997
```

Le point fixe est clarement hors du domaine.

Détermine la valeur de la fonction au point fixe

```
Ys = ao+.5*Xs'*a;
```

Détermine le valeurs et vecteurs propres

```
[V,D] = eig(A);
disp('Les valeurs propres')
```

```
Les valeurs propres
```

```
disp(diag(D))
```

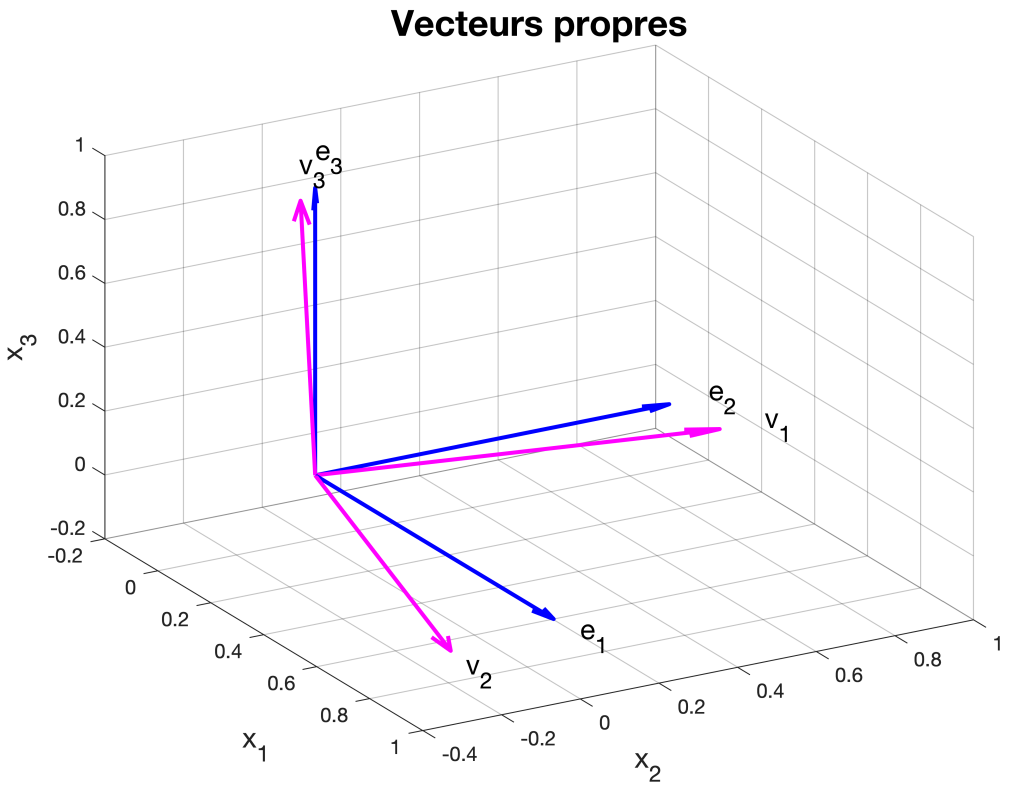
```
   -0.2066
   -0.0150
    0.0005
```

Le fait d'avoir une valeur propre λ_3 négative alors que les deux autres, λ_2 et λ_2 soient positives révelle que la géométrie est hyperbolique. Cependant, le point fixe étant très éloigné du domaine, cette géométrie hyperbolique n'est pas visible, et d'autant plus que λ_3 est très petite par rapport avec les deux autres.

On peut dessiner les axes d'après les vecteurs propres:

```
% origine
eo=zeros(3,1);

% vecteur de base
ex=[1;0;0];
ey=[0;1;0];
ez=[0;0;1];

% vecteur propre
u=-V(1,:)';
v=-V(2,:)';
w=V(3,:)';
```

```
figure
quiver3(eo,eo,eo,ex,ey,ez,"LineWidth",2,'Color','blue')
hold on
quiver3(eo,eo,eo,u,v,w,"LineWidth",2,'Color','magenta')
hold off
title('Vecteurs propres','FontSize',18)
xlabel('x_1','FontSize',14)
ylabel('x_2','FontSize',14)
zlabel('x_3','FontSize',14)

% légendes
text(1,0,0,'e_1','FontSize',14)
text(0,1,0,'e_2','FontSize',14)
text(0,0,1,'e_3','FontSize',14)
text(u,v,w,{'v_1';'v_2';'v_3'},'FontSize',14)

view([60 30])
```



On observe que les vecteurs propres sont assez proches des vecteurs de base.

Visualisation 3D

- *Définition des points sur lesquels la fonction sera évaluée*

```
[X1,X2,X3]=meshgrid(-1:.2:1); %
```

- Calcul de la fonction aux points du réseau

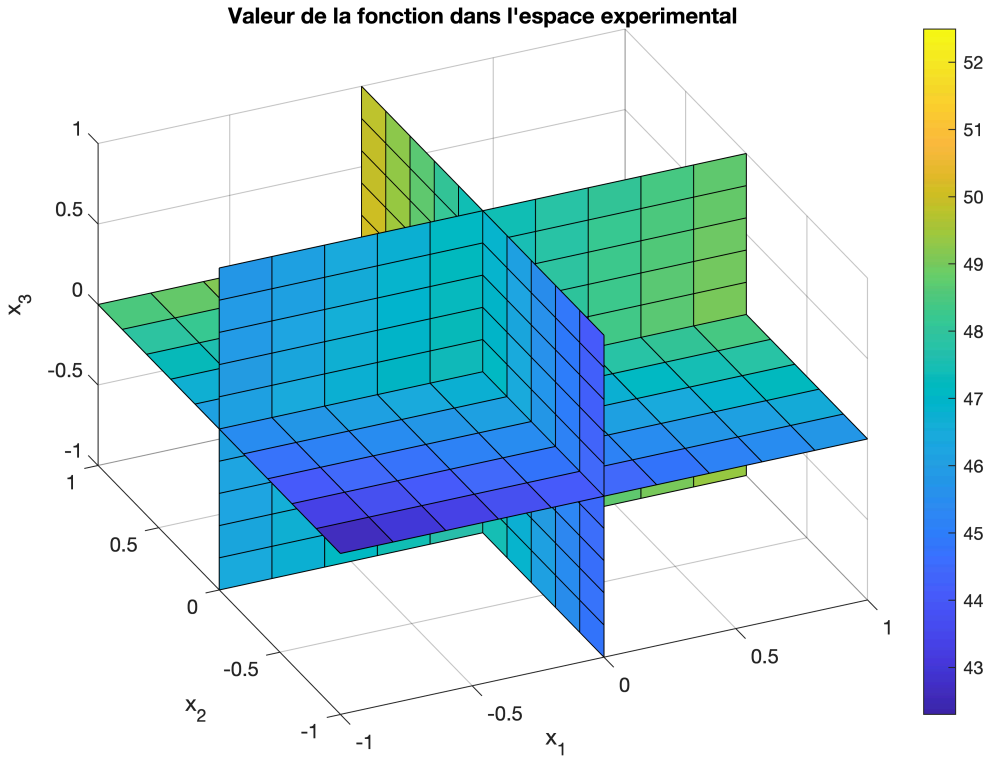
```
Y_est=ao+ a(1)*X1 + a(2)*X2 + a(3)*X3 +...
    A(1,2)*2*X1.*X2 +A(1,3)*2*X1.*X3+A(2,3)*2*X2.*X3+...
    A(1,1)*X1.^1 +A(2,2)*X2.^2 + A(3,3)*X3.^2;
```

- Les plans passent par le point central (le point fixe est hors du domaine)

```
xslice=0;
yslice=0;
zslice= 0;
```

- Dessin des plans

```
figure
slice(X1,X2,X3,Y_est,xslice,yslice,zslice)
title('Valeur de la fonction dans l''espace experimental')
xlabel('x_1')
ylabel('x_2')
zlabel('x_3')
colorbar
map=colormap;
caxis('manual')
view([-24.70 40.40])
```

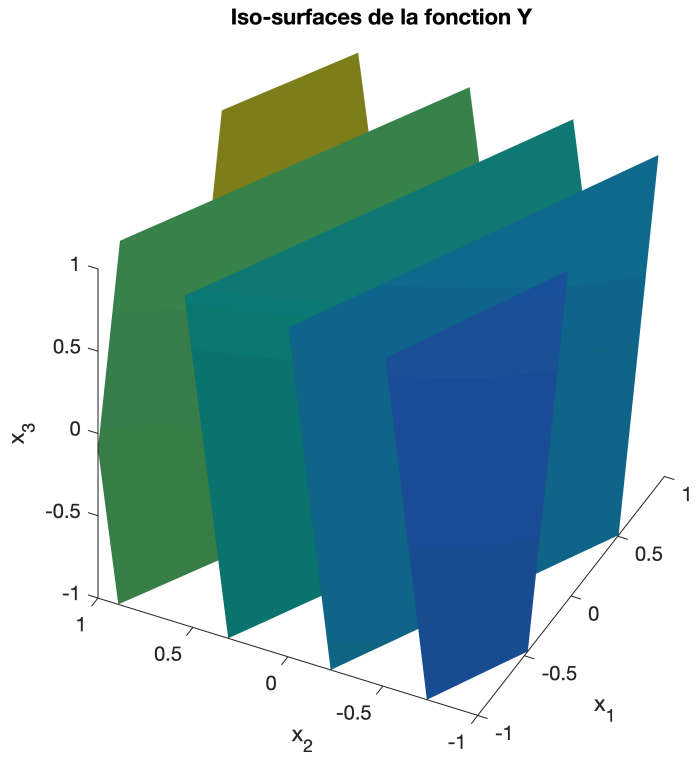


- Mémorise l'échelle des couleurs pour la suite

```
Ncouleur=size(map,1);
[Cmin,Cmax]=caxis;
```

- Tracer des isosurfaces

```
figure
Nsurf=5;
valeur= linspace(min(min(min(Y_est))),max(max(max(Y_est))),Nsurf+2);
for k=2:Nsurf+1
    p=patch(isosurface(X1,X2,X3,Y_est,valeur(k)));
    isonormals(X1,X2,X3,Y_est,p)
    p.FaceColor=map(10+k*30,1:3);
    p.EdgeColor='none';
    hold on
end
hold off
daspect([1 1 1])
view(3);
axis tight
camlight
lighting gouraud
view([-48.30 8.40])
title('Iso-surfaces de la fonction Y')
xlabel('x_1')
ylabel('x_2')
zlabel('x_3')

view([-63.82 38.98])
```




```
function S=dessin_3D(X,az,el,titre)
% * Chaque ligne correspond à un point de l'espace expérimental
figure
plot3(X(:,1),X(:,2),X(:,3),'ok','MarkerSize',12,'MarkerFaceColor','r')
axis([-1 1 -1 1 -1 1])

view(az,el) % azimut and elevation du point de vue
title(titre,'FontName','Times New Roman','FontSize',20,'FontWeight','bold')
xlabel('Axe 1','FontName','Times New Roman','FontSize',14,'FontWeight','bold')
ylabel('Axe 2','FontName','Times New Roman','FontSize',14,'FontWeight','bold')
zlabel('Axe 3','FontName','Times New Roman','FontSize',14,'FontWeight','bold')
hold on
        %
        % * Dessiner un repère xyz
        quiver3(0,0,0,1,0,0,'LineWidth',3)
        text(.9,.1,.1,'e_1','FontName','Times New Roman','FontSize',14,'FontWeight','bold')
        quiver3(0,0,0,0,1,0,'LineWidth',3)
        text(.1,.9,.1,'e_2','FontName','Times New Roman','FontSize',14,'FontWeight','bold')
        quiver3(0,0,0,0,0,1,'LineWidth',3)
        text(.1,.1,.9,'e_3','FontName','Times New Roman','FontSize',14,'FontWeight','bold')
        %
        % * Créer un polygone à partir des points d'expériences
        S.Vertices = X;
        S.Faces = convhull(X);
        % il y a plusieurs fonctions possibles
        % bundary(), alphaShape()
        S.FaceVertexCData = [0;1;2;3;0;1;2;3;0;1;2;3;0;1;2;3;0;1;2;3];
```

```
        S.FaceColor = 'flat';
        S.EdgeColor = 'black';
        S.FaceAlpha= .8;
        patch(S)
        box on
        grid on
hold off
text(X(:,1),X(:,2),X(:,3),num2cell(1:13),"FontSize",20)
end
```