Series 9: Response surface

Table of Contents

Remark

Since the simulations of the measurement have a random component, there may be differences between the figures and the direct calculations.

A. Factorial design 3^3

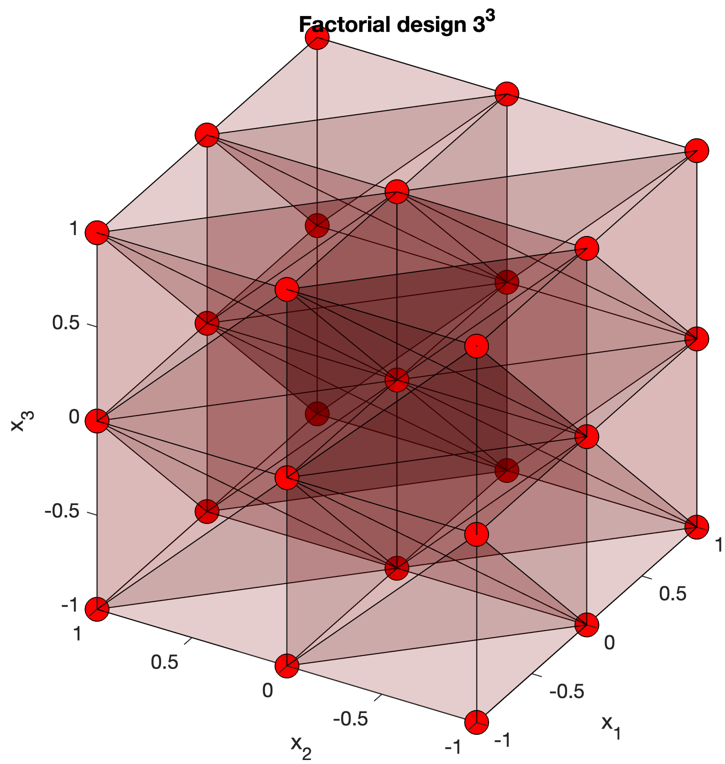
Step1: Generation of the design of 27 runs

The Matlab routine *fullfact()* let us define the 3^3 factorial matrix of experiments with three levels for each factor.

```
E1=fullfact([3 3 3])-2;
```

The user-defined routine *axonometry(E,title) draws* the distribution of the points in a 3D space. We can onserve that it constitutes a periodic mesh of points.

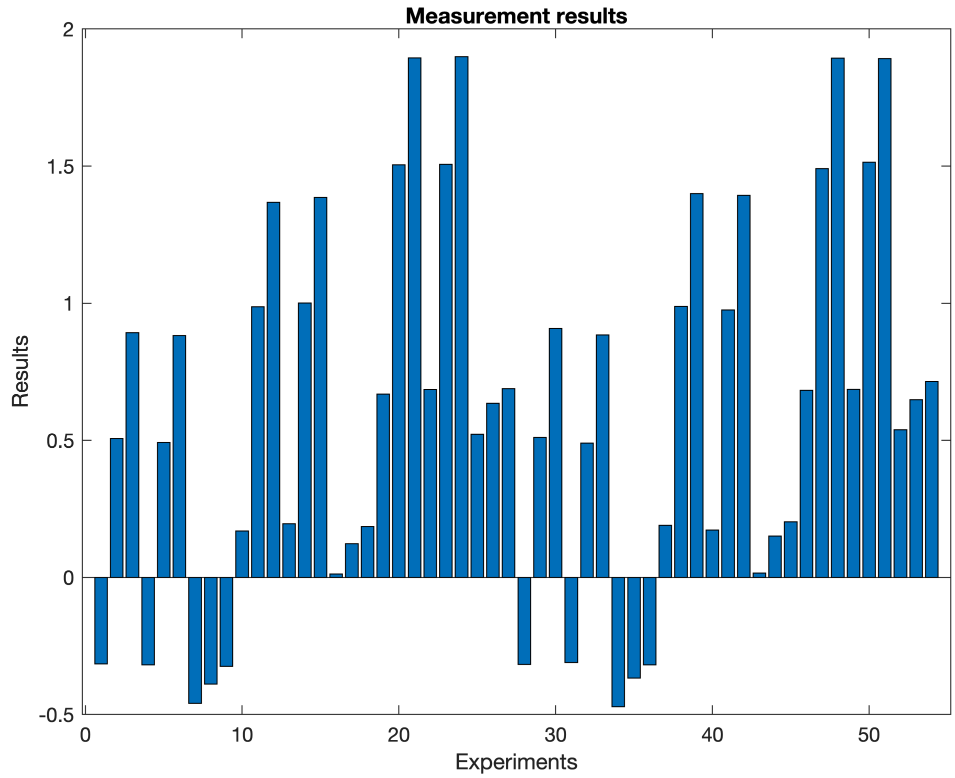
```
axonometry(E1,'Factorial design 3^3');
```



A.1 Simulating the measurements

The user-defined routine *mesure3(E)* provides the results of a virtual experiment modelized with a response function and a normaly distributed error. The routine is provided with the statement of the exercise. The measurement is done two times, constituting two replicates. The figure below presents the measurement results as a bar chart.

```
Y1=mesure3([E1;E1]); % the measurement is done 2 times
figure
bar(Y1)
title('Measurement results')
xlabel('Experiments')
ylabel('Results')
```



A.2 Fitting a quadratic model

A quadratic model is fitted with the Matlab routine *fitlm().* Estimates and p-values corresponding to a Student test are given in the output. Coefficients a_{13}, a_{23} and a_{33} have values largely higher than the standard threshold of 5%. It is also possible to observe that the standard error of those coefficients are an order of magnitude larger than the estimates.

```
mdl1=fitlm([E1;E1], Y1,'quadratic')
```

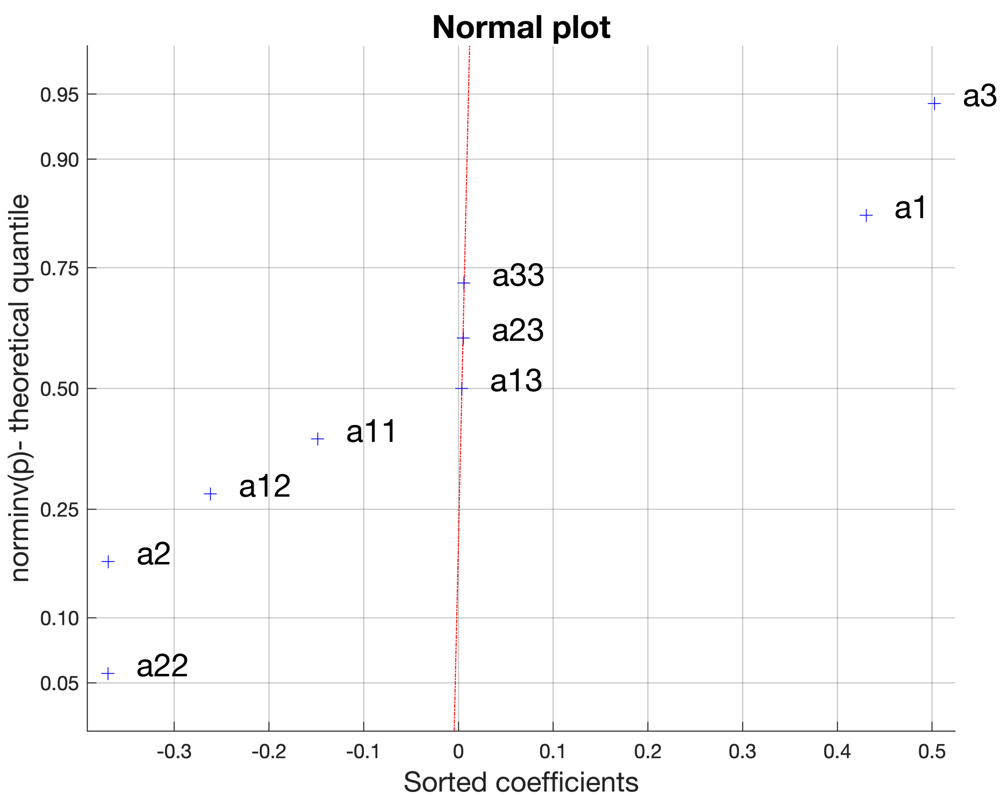
```
mdl1 =
Linear regression model:
    y ~ 1 + x1*x2 + x1*x3 + x2*x3 + x1^2 + x2^2 + x3^2

Estimated Coefficients:
```

	Estimate	SE	tStat	pValue
(Intercept)	0.95209	0.043002	22.141	1.7455e-25
x1	0.43035	0.019906	21.619	4.5702e-25
x2	-0.36986	0.019906	-18.58	1.8798e-22
x3	0.50251	0.019906	25.244	8.2169e-28
x1:x2	-0.26196	0.02438	-10.745	6.9522e-14
x1:x3	0.0034863	0.02438	0.143	0.88694
x2:x3	0.0052281	0.02438	0.21444	0.83119
x1^2	-0.14858	0.034478	-4.3093	9.0646e-05

```
    x2^2            -0.37014     0.034478     -10.735    7.1559e-14
    x3^2           0.0057393     0.034478     0.16646       0.86856


Number of observations: 54, Error degrees of freedom: 44
Root Mean Squared Error: 0.119
R-squared: 0.975,  Adjusted R-Squared: 0.97
F-statistic vs. constant model: 189, p-value = 3.88e-32
```

A user-defined routine draws a normal plot discriminating the problematic coefficients .

```
normplot_DOE(mdl1.Coefficients.Estimate(2:end),5:7,-0.2, 0.2,0.03,0.05)
```



The neglected coefficients are well aligned at the center of the plot and can be reasonably considered as due to the noise in the measurement. The retained coefficients are therefore a_1, a_2, a_3, a_{12}, a_{11} and a_{22} . The model is $\hat{y} = a_o + a_1x_1 + a_2x_2 + a_3x_3 + a_{12}x_1x_2 + a_{11}x_1^2 + a_{22}x_2^2$. It is now possible to define a matrix with the terms of the final model and to proceed to the regression of the model with only the significant coefficients :

```
spec=[0  0  0
    1  0  0
    0  1  0
    0  0  1
    1  1  0
    2  0  0
    0  2  0];

mdl_fact=fitlm([E1;E1],Y1,spec)
```

```
mdl_fact =
```

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

Estimated Coefficients:
                   Estimate       SE        tStat       pValue  
                   ________    ________    _______    __________

    (Intercept)     0.95591    0.035202     27.155    2.2335e-30
    x1              0.43035    0.019281      22.32    1.1691e-26
    x2             -0.36986    0.019281    -19.183    7.2327e-24
    x3              0.50251    0.019281     26.062    1.3684e-29
    x1:x2          -0.26196    0.023614    -11.094    1.0123e-14
    x1^2           -0.14858    0.033395     -4.449    5.2687e-05
    x2^2           -0.37014    0.033395    -11.084    1.0439e-14


Number of observations: 54, Error degrees of freedom: 47
Root Mean Squared Error: 0.116
R-squared: 0.975,  Adjusted R-Squared: 0.971
F-statistic vs. constant model: 302, p-value = 8.8e-36
```

```
anova(mdl_fact,'summary',2)
```

ans = 7×5 table

	SumSq	DF	MeanSq	F	pValue
1 Total	24.8671	53	0.4692	NaN	NaN
2 Model	24.2381	6	4.0397	301.8530	8.7987e-36
3 . Linear	20.6822	3	6.8941	515.1381	6.0893e-36
4 . Nonlinear	3.5559	3	1.1853	88.5680	2.3425e-19
5 Residual	0.6290	47	0.0134	NaN	NaN
6 . Lack of fit	0.6259	20	0.0313	268.6505	5.2953e-26
7 . Pure error	0.0031	27	0.0001	NaN	NaN

By inspecting the outputs of the routine, it is possible to observe that all the p-values are small enough. The biggest is $7 \cdot 10^{-5}$. Nevertheless, the **analysis of variance** makes it possible to detect a **lack of fit.** The model would therefore deserve to be completed. In the case of this exercise, we do not dig further. But if necessary it would be the analysis of the residues that would give some clues on the corrections to bring to the model.

A.3 Canonical analysis

The model does not include any cross or cubic terms involving the factor x_3, so we can perform the canonical analysis only on x_1 and x_2 , fixing $x_3 = 0$. Let's compute the vector of linear coefficients, *a*, and the matrix of coefficients of the second degree, *A*:

```
coef=mdl_fact.Coefficients.Estimate;
ao=coef(1);
a= coef(2:3);
```

```
A= [coef(6) coef(5)/2;coef(5)/2 coef(7)];
```

a_o:

```
disp(ao)
```

```
    0.9559
```

a:

```
disp(a)
```

```
    0.4303
   -0.3699
```

A:

```
disp(A)
```

```
   -0.1486   -0.1310
   -0.1310   -0.3701
```

Computing the fixed point of vectors and the eigenvectors, the eigenvalues and the angle between the two frames:

```
xs= -inv(A)*a/2 % fixed point
```

```
xs = 2×1
    2.7451
   -1.4710
```

Take note that the fixed point is placed outside of the experimental space.

```
ys=ao+a'*xs+xs'*A*xs % value of the function at the fixed point
```

```
ys = 1.8186
```

```
[V,L]=eig(A) % eigen vectors and eigen values
```

```
V = 2×2
    0.4208   -0.9071
    0.9071    0.4208
L = 2×2
   -0.4309         0
         0   -0.0878
```

```
teta=atan(V(2,1)/V(1,1))/pi*180 % angle between the two frames
```

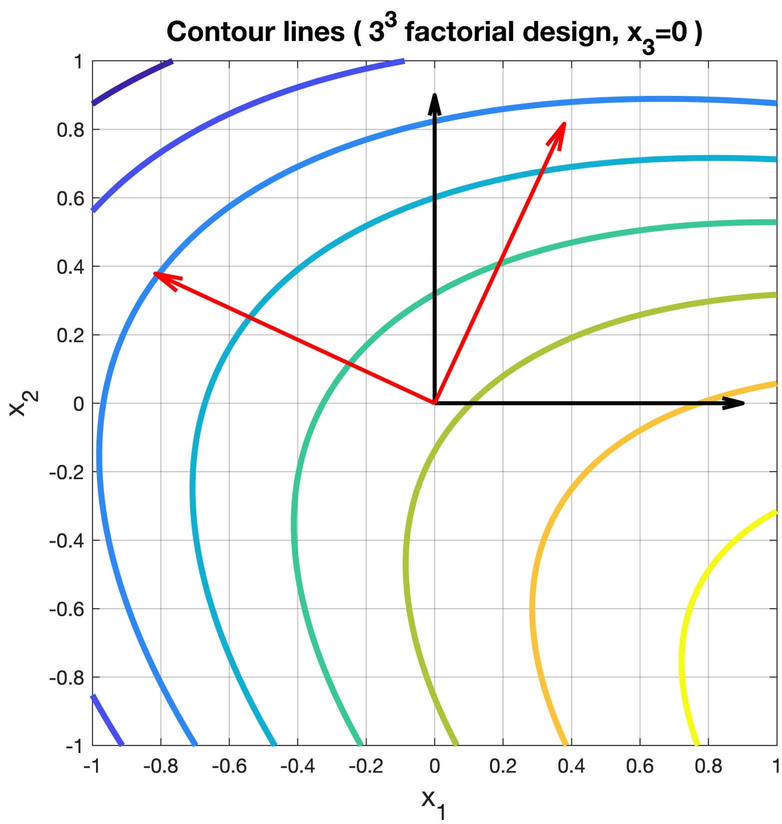
```
teta = 65.1119
```

The two eigenvalues of the same sign reveal an elliptical surface response. This function decreases as one moves away from the fixed point $(2.8, -1.47)$. The canonical form is

$$Y = a_o + \sum \lambda_i \widetilde{X}_i = 1.84 - 0.44\widetilde{X}_1^2 - 0.09\widetilde{X}_2^2$$

The figure below shows a slice of the model at $x_3 = 0$. The contour lines of the fitted function can be represented easily thanks to the symbolic functions available on Matlab and reveal also the ellictic geometry of the solution. The two frames have been placed in the figure at the coordinates (0,0,0). The new axes are $\widetilde{X}_1 = 0.41x_1 + 0.91x_2$ et $\widetilde{X}_2 = -0.91x_1 + 0.41x_2$. They have an angle of 66° with the original frame.

```
syms x1 x2
y1=ao+ a(1)*x1+a(2)*x2+ 2*A(1,2)*x1*x2+ A(1,1)*x1^2+ A(2,2)*x2^2;
figure
fcontour(y1,[-1 1 -1 1],'LineWidth',3)
grid on
pbaspect([1 1 1])
title('Contour lines ( 3^3 factorial design, x_3=0 )', 'FontSize',14)
xlabel('x_1', 'FontSize',14)
ylabel('x_2', 'FontSize',14)
hold on
quiver([0,0],[0,0],[1 0],[0,1],'k-','LineWidth',2)
quiver([0,0],[0,0],V(1,:),V(2,:),'r-','LineWidth',2)
hold off
```



That's all for the moment for this design.

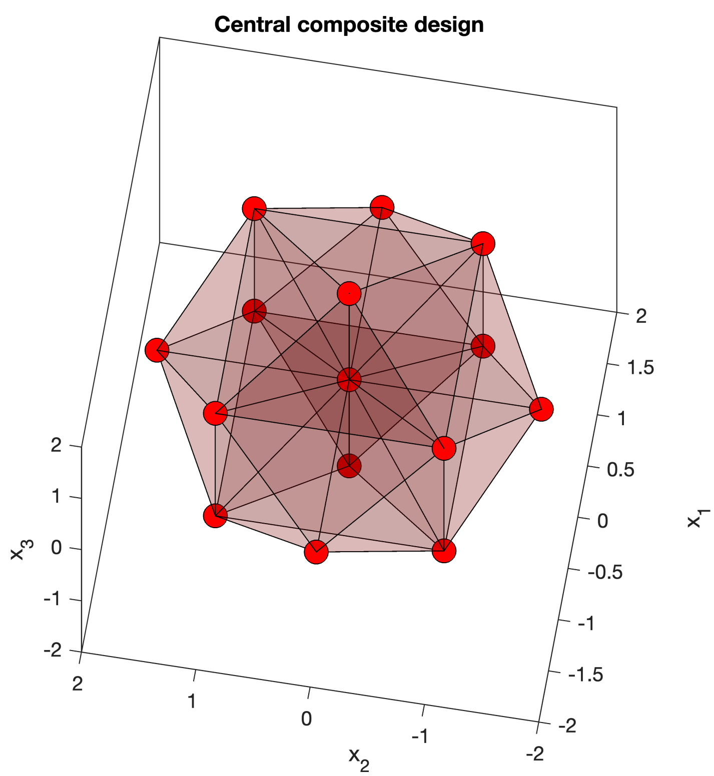
B. Central composite design

The central composite design is an interesting option to built a model step by step by increasing the degree of the polynome. Let's start by generating the design of 15 runs with the Matlab routine *ccdesign().* We can observe the spherical distribution of the points

```
E2= ccdesign(3, 'center',1);
```

The routine axonometry let us represent the distribution of the points in a 3D space.

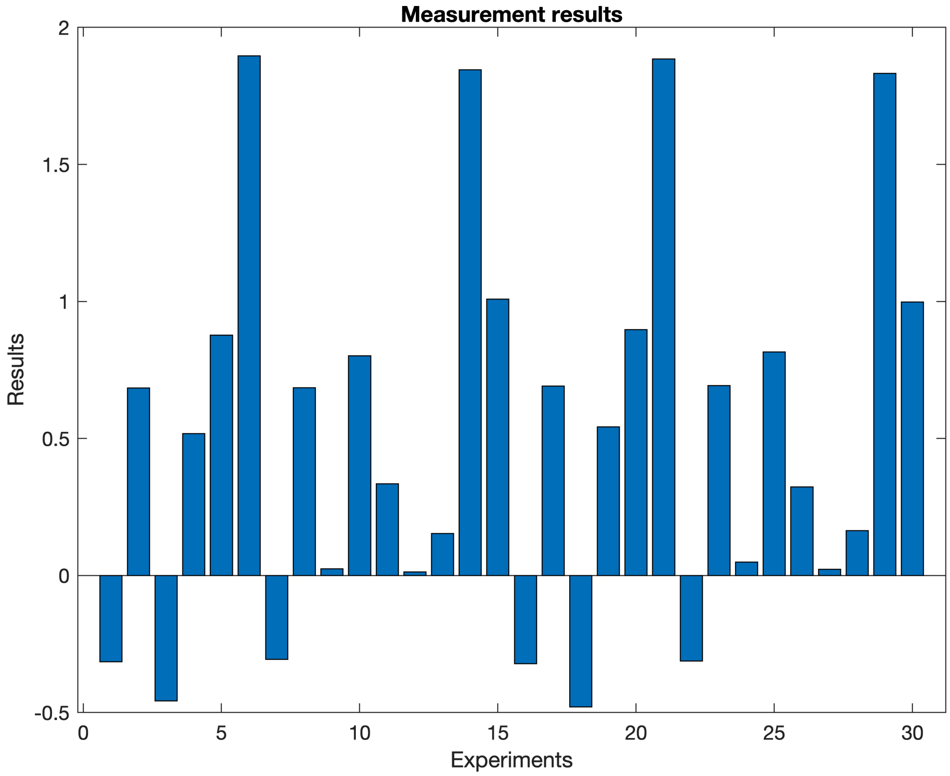
```
axonometry(E2,'Central composite design');
view([-80.30 63.75])
```



B.1 Simulating the measurement

The user-defined routine *mesure3(E)* provides the results of a virtual experiment modelized with a response function and a normaly distributed error. The routine is provided with the statement of the exercise. The measurement is done two times, constituting two replicates. The figure below presents the measurement results as a bar chart.

```
Y2=mesure3([E2;E2]); % The measurement is done two times
figure
bar(Y2)
title('Measurement results')
xlabel('Experiments')
ylabel('Results')
```



B.2 Fitting a quadratic model

A quadratic model is fitted with the Matlab routine *fitlm().* Estimates and p-values corresponding to a Student test are given in the output. AS previously, coefficients a_{13}, a_{23} and a_{33} have values largely higher than the standard threshold of 5%. It is also possible to observe that the standard error of those coefficients are an order of magnitude larger than the estimates.

```
mdl2=fitlm([E2;E2], Y2,'quadratic')
```

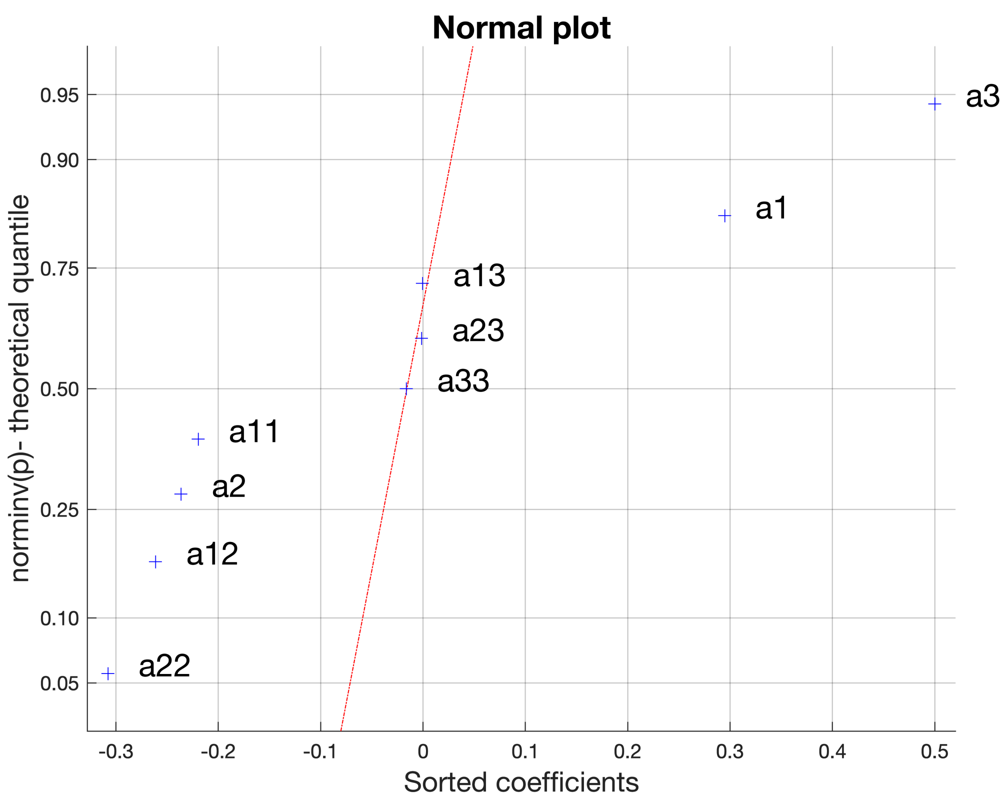
```
mdl2 =
Linear regression model:
    y ~ 1 + x1*x2 + x1*x3 + x2*x3 + x1^2 + x2^2 + x3^2

Estimated Coefficients:
```

	Estimate	SE	tStat	pValue
(Intercept)	1.0138	0.11143	9.0983	1.5121e-08
x1	0.29474	0.03033	9.7178	5.1087e-09
x2	-0.23641	0.03033	-7.7947	1.7377e-07
x3	0.50018	0.03033	16.491	4.1401e-13
x1:x2	-0.26124	0.039628	-6.5923	2.0181e-06
x1:x3	-0.00036327	0.039628	-0.0091671	0.99278
x2:x3	-0.0014878	0.039628	-0.037544	0.97042
x1^2	-0.21968	0.045558	-4.8219	0.00010361
x2^2	-0.30785	0.045558	-6.7573	1.4249e-06
x3^2	-0.016122	0.045558	-0.35387	0.72714

```
Number of observations: 30, Error degrees of freedom: 20
Root Mean Squared Error: 0.159
R-squared: 0.965,  Adjusted R-Squared: 0.949
F-statistic vs. constant model: 61, p-value = 1.41e-12
```

Selecting of significant coefficients (proceed iteratively: produce the plot with some neglected coefficients in the center of the list and then refined)

```
normplot_DOE(mdl2.Coefficients.Estimate(2:end),5:7,-0.1, 0.1,0.03,0.05)
```



The retained coefficients are therefore a_1, a_2, a_3, a_{12}, a_{11} and a_{22} which makes it possible to define the matrix of the terms of the model.

```
spec=[0  0  0
     1  0  0
     0  1  0
```

```
     0  0  1
     1  1  0
     2  0  0
     0  2  0];
```

The model is therefore $\hat{y} = a_o + a_1x_1 + a_2x_2 + a_3x_3 + a_{12}x_1x_2 + a_{11}x_1^2 + a_{22}x_2^2$

The regression for this model is

```
mdl_CC=fitlm([E2;E2],Y2,spec)
```

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

Estimated Coefficients:
                   Estimate       SE        tStat       pValue  
                   ________    ________    _______    __________

    (Intercept)      0.9809    0.057345     17.105    1.4167e-14
    x1              0.29474    0.028372     10.388    3.7076e-10
    x2             -0.23641    0.028372    -8.3325    2.1177e-08
    x3              0.50018    0.028372     17.629    7.4226e-15
    x1:x2          -0.26124     0.03707    -7.0471    3.5163e-07
    x1^2           -0.20966    0.033381    -6.2808    2.0789e-06
    x2^2           -0.29783    0.033381    -8.9222    6.2797e-09


Number of observations: 30, Error degrees of freedom: 23
Root Mean Squared Error: 0.148
R-squared: 0.965,  Adjusted R-Squared: 0.955
F-statistic vs. constant model: 104, p-value = 1.62e-15
```

```
anova(mdl_CC,'summary',2)
```

ans = 7×5 table

	SumSq	DF	MeanSq	F	pValue
1 Total	14.2916	29	0.4928	NaN	NaN
2 Model	13.7859	6	2.2976	104.4992	1.6176e-15
3 . Linear	10.7327	3	3.5776	162.7117	1.2563e-15
4 . Nonlinear	3.0531	3	1.0177	46.2866	6.6208e-10
5 Residual	0.5057	23	0.0220	NaN	NaN
6 . Lack of fit	0.5041	8	0.0630	595.8691	2.3454e-17
7 . Pure error	0.0016	15	0.0001	NaN	NaN

By inspecting the result of the routine, we observe that all the p-values are small enough. The biggest is $2 \cdot 10^{-6}$.

The analysis of variance makes it possible to detect a lack of fit, the model deserves to be completed.

B.3 Canonical analysis

The model does not include any cross or cubic term involving the factor x_3, so we can perform the canonical analysis only on x_1 and x_2 , fixing $x_3 = 0$.

Calculating the vector of linear coefficients and the matrix of coefficients of the second degree:

```
coef=mdl_CC.Coefficients.Estimate;
ao=coef(1);
a= coef(2:3);
A= [coef(6) coef(5)/2;coef(5)/2 coef(7)];
```

a_o:

```
disp(ao)
```

```
    0.9809
```

a:

```
disp(a)
```

```
    0.2947
   -0.2364
```

A:

```
disp(A)
```

```
   -0.2097   -0.1306
   -0.1306   -0.2978
```

Computing the fixed point and the eigenvalues:

```
xs= -inv(A)*a/2 % Fixed point
```

```
xs = 2×1
    1.3074
   -0.9703
```

```
ys=ao+a'*xs+xs'*A*xs % Value of the function at the fixed point
```

```
ys = 1.2883
```

```
[V,L]=eig(A) % eigenvectors and eigenvalues
```

```
V = 2×2
    0.5832   -0.8123
    0.8123    0.5832
L = 2×2
   -0.3916         0
         0   -0.1159
```

```
teta=atan(V(2,1)/V(1,1))/pi*180 % Angle between the two frames
```

```
teta = 54.3253
```

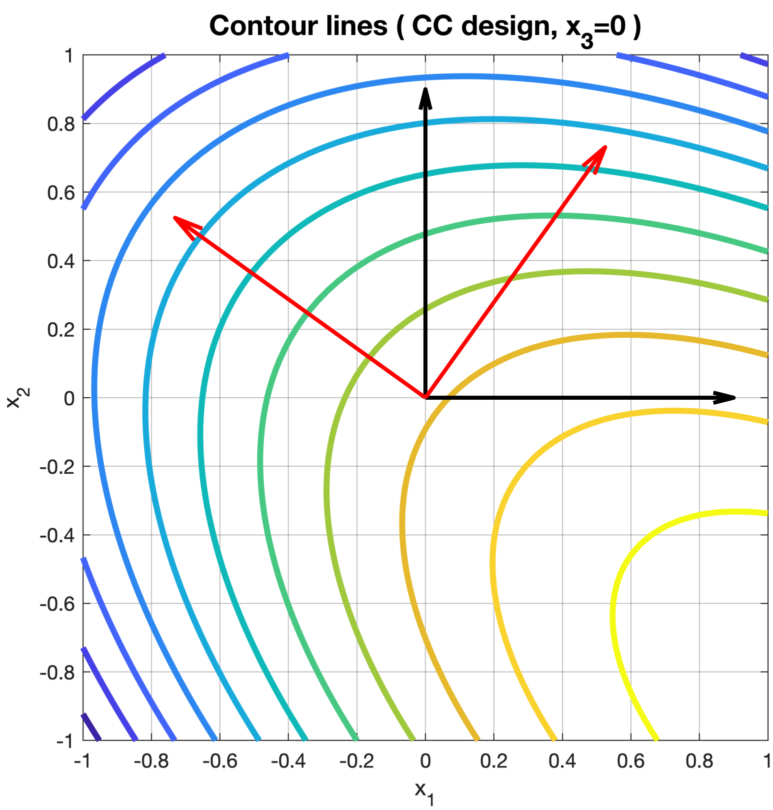
The two eigenvalues of the same sign reveal an elliptical surface. This function decreases as one moves away from the fixed point $(1.3,-1)$. The canonical form is

$$Y = a_o + \sum \lambda_i \widetilde{X}_i = 1.29 - 0.40\widetilde{X}_1^2 - 0.12\widetilde{X}_2^2$$

The new axes are $\widetilde{X}_1 = 0.57x_1 + 0.82x_2$ et $\widetilde{X}_2 = -0.82x_1 + 0.57x_2$. They form an angle of 54° with the original frame.

The function can be represented easily with the symbolic functions available on Matlab

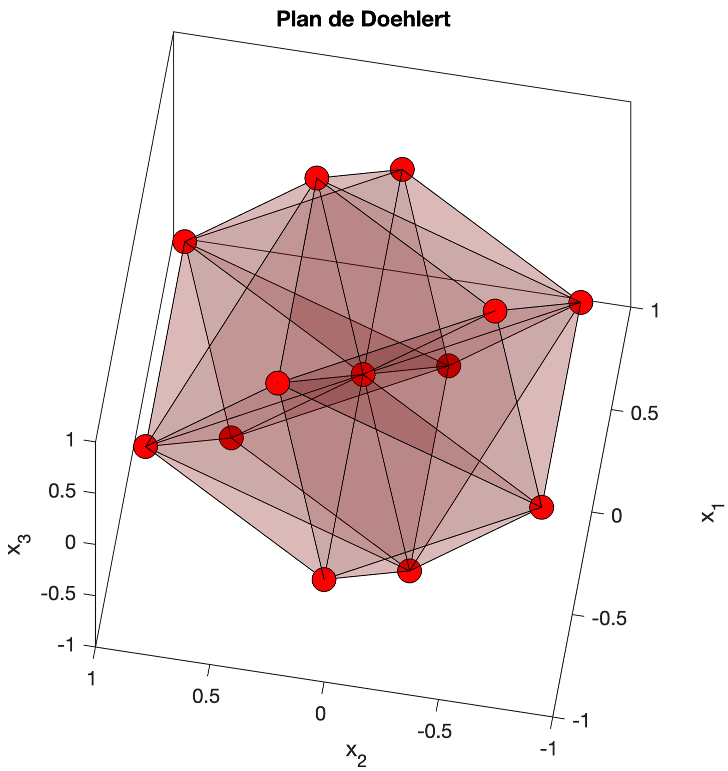
```
syms x1 x2
y2=ao+ a(1)*x1+a(2)*x2+ 2*A(1,2)*x1*x2+ A(1,1)*x1^2+ A(2,2)*x2^2;
figure
fcontour(y2,[-1 1 -1 1],'LineWidth',3)
title('Contour lines ( CC design, x_3=0 )', 'FontSize',14)
xlabel('x_1')
ylabel('x_2')
pbaspect([1 1 1])
grid on
hold on
quiver([0,0],[0,0],[1 0],[0,1],'k-','LineWidth',2)
quiver([0,0],[0,0],V(1,:),V(2,:),'r-','LineWidth',2)
hold off
```



C Doehlert's design

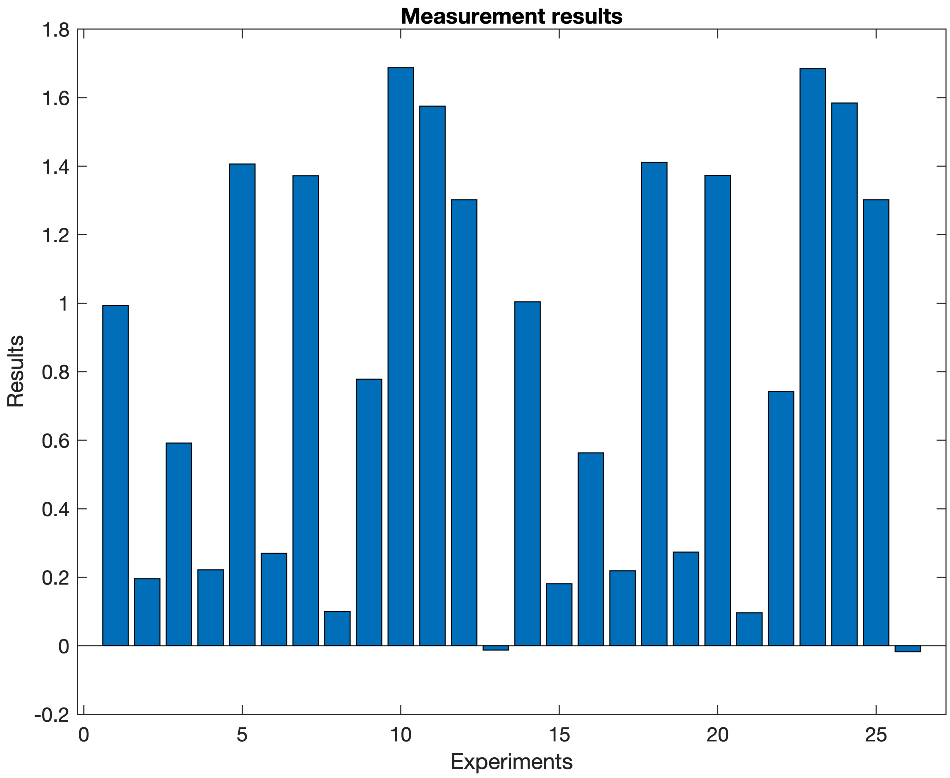
Generating the design of 13 runs Doehlert design

```
E3= doehlert(3);
axonometry(E3,'Plan de Doehlert');
view([-80.30 63.75])
```



C.1 Simulating the measurement

```
Y3=mesure3([E3;E3]); % two replicates
figure
bar(Y3)
title('Measurement results')
xlabel('Experiments')
ylabel('Results')
```



C.2 Fitting a quadratic model

```
mdl3=fitlm([E3;E3], Y3,'quadratic')
```

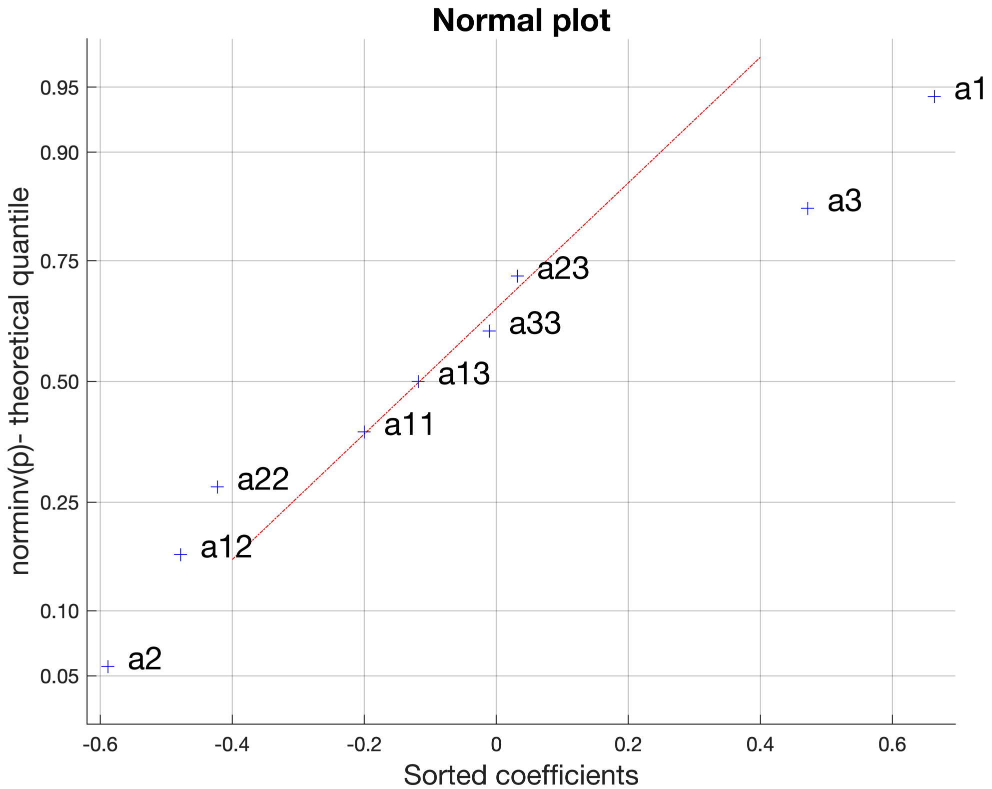
```
mdl3 =
Linear regression model:
    y ~ 1 + x1*x2 + x1*x3 + x2*x3 + x1^2 + x2^2 + x3^2

Estimated Coefficients:
```

	Estimate	SE	tStat	pValue
(Intercept)	0.99868	0.082478	12.108	1.8058e-09
x1	0.66383	0.041239	16.097	2.6391e-11
x2	-0.58844	0.041239	-14.269	1.6124e-10
x3	0.47165	0.041239	11.437	4.1149e-09
x1:x2	-0.47831	0.095238	-5.0223	0.00012513
x1:x3	-0.11834	0.10648	-1.1114	0.28282
x2:x3	0.031894	0.10648	0.29953	0.76839
x1^2	-0.20008	0.10101	-1.9807	0.065079
x2^2	-0.42252	0.10101	-4.1828	0.0007034
x3^2	-0.010836	0.098169	-0.11039	0.91348

```
Number of observations: 26, Error degrees of freedom: 16
Root Mean Squared Error: 0.117
R-squared: 0.976,  Adjusted R-Squared: 0.963
F-statistic vs. constant model: 73, p-value = 2.76e-11
```

Selecting the significant coefficients

```
normplot_DOE(mdl3.Coefficients.Estimate(2:end),4:7,-0.4, 0.4,0.03,0.05)
```



The retained coefficients are a_1, a_2, a_3, a_{12} and a_{22}. This let us define the matrix of coefficients *spec*

```
spec=[0 0 0
     1 0 0
     0 1 0
     0 0 1
     1 1 0
     2 0 0
     0 2 0];
```

The model is therefore $\hat{y} = a_o + a_1x_1 + a_2x_2 + a_3x_3 + a_{12}x_1x_2 + a_{11}x_1^2 + a_{22}x_2^2$

The coefficient a_{11} has been retained even if it was at the limit of validity ...this implies that it better to follow it significance in the rest of the analysis

Let's proceed to the inference of this specific model

```
mdl_doe=fitlm([E3;E3],Y3,spec)
```

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

Estimated Coefficients:
```

```
                 Estimate       SE         tStat       pValue
                 ________    ________    _______    __________

    (Intercept)    0.99104    0.042728     23.194    2.1206e-15
    x1             0.66383    0.039393     16.851    6.9889e-13
    x2            -0.58844    0.039393    -14.938    5.9224e-12
    x3             0.47165    0.039393     11.973    2.6922e-10
    x1:x2         -0.51178    0.086306    -5.9298    1.0425e-05
    x1^2          -0.18856    0.073636    -2.5607      0.019118
    x2^2          -0.42002    0.073636     -5.704    1.6922e-05


Number of observations: 26, Error degrees of freedom: 19
Root Mean Squared Error: 0.111
R-squared: 0.974,  Adjusted R-Squared: 0.966
F-statistic vs. constant model: 120, p-value = 4.6e-14
```

```
anova(mdl_doe,'summary',2)
```

ans = 7×5 table

	SumSq	DF	MeanSq	F	pValue
1 Total	9.1597	25	0.3664	NaN	NaN
2 Model	8.9238	6	1.4873	119.8023	4.6031e-14
3 . Linear	8.0751	3	2.6917	216.8166	7.1791e-15
4 . Nonlinear	0.8487	3	0.2829	22.7879	1.6435e-06
5 Residual	0.2359	19	0.0124	NaN	NaN
6 . Lack of fit	0.2345	6	0.0391	372.8019	8.9284e-14
7 . Pure error	0.0014	13	0.0001	NaN	NaN

By inspecting the output of the routine, we observe that all the p-values are sufficiently small, except for the term a_{11}which has a p-value of 2%. The Doehlert design seems to have more difficulty than the other designs in detecting curvature in the direction x_1. The analysis of variance makes it possible to detect a lack of fit, the model deserves to be completed.

C.3 Canonical analysis :

The model does not include any cross or cubic term involving the factor x_3, so we can perform the canonical analysis only on $x_1$and x_2 , fixing $x_3 = 0$.

Calculating the vector of linear coefficients and the matrix of coefficients of the second degree:

```
coef=mdl_doe.Coefficients.Estimate;
ao=coef(1);
a= coef(2:3);
A= [coef(6) coef(5)/2;coef(5)/2 coef(7)];
```

a_o:

```
disp(ao)
```

```
    0.9910
```

a:

```
disp(a)
```

```
    0.6638
   -0.5884
```

A:

```
disp(A)
```

```
   -0.1886   -0.2559
   -0.2559   -0.4200
```

Computing the fixed point and the eigenvalues:

```
xs= -inv(A)*a/2 % fixed point
```

```
xs = 2×1
   15.6485
  -10.2341
```

```
ys=ao+a'*xs+xs'*A*xs % value of the function at the fixed point
```

```
ys = 9.1961
```

```
[V,L]=eig(A) % Eigenvectors and eigenvalues
```

```
V = 2×2
    0.5422   -0.8403
    0.8403    0.5422
L = 2×2
   -0.5851         0
         0   -0.0234
```

```
teta=atan(V(2,1)/V(1,1))/pi*180 % Angle between the two frames
```

```
teta = 57.1676
```

The two eigenvalues of the same sign reveal an elliptical surface. This function decreases as one moves away from the fixed point $(12.5, -8.5)$. The canonical form is

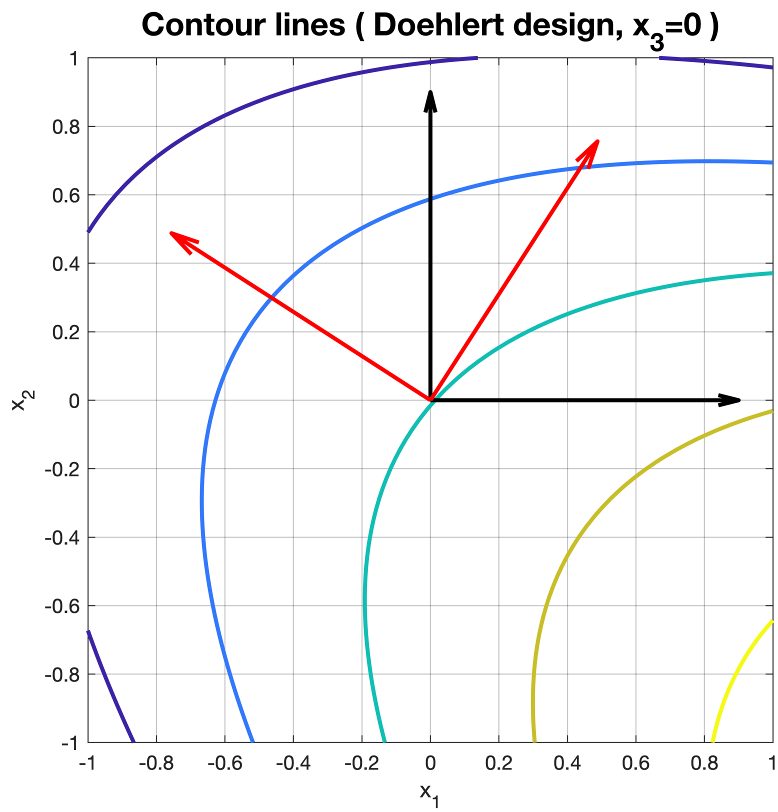
$$Y = a_o + \sum \lambda_i \widetilde{X}_i = 7.66 - 0.59\widetilde{X}_1^2 - 0.03\widetilde{X}_2^2$$

The axes are $\widetilde{X}_1 = 0.56x_1 + 0.83x_2$ and $\widetilde{X}_2 = -0.83x_1 + 0.56x_2$. The angle between the two frames is 56°.

As the problem can be viewed in two dimension, a contourline diagram can be produced:

```
syms x1 x2
```

```
y3=ao+ a(1)*x1+a(2)*x2+ 2*A(1,2)*x1*x2+ A(1,1)*x1^2+ A(2,2)*x2^2;
figure
fcontour(y3,[-1 1 -1 1],'LineWidth',2)
title('Contour lines ( Doehlert design, x_3=0 )', 'FontSize',16)
xlabel('x_1')
ylabel('x_2')
pbaspect([1 1 1])
grid on
hold on
quiver([0,0],[0,0],[1 0],[0,1],'k-','LineWidth',2)
quiver([0,0],[0,0],V(1,:),V(2,:),'r-','LineWidth',2)
hold off
```



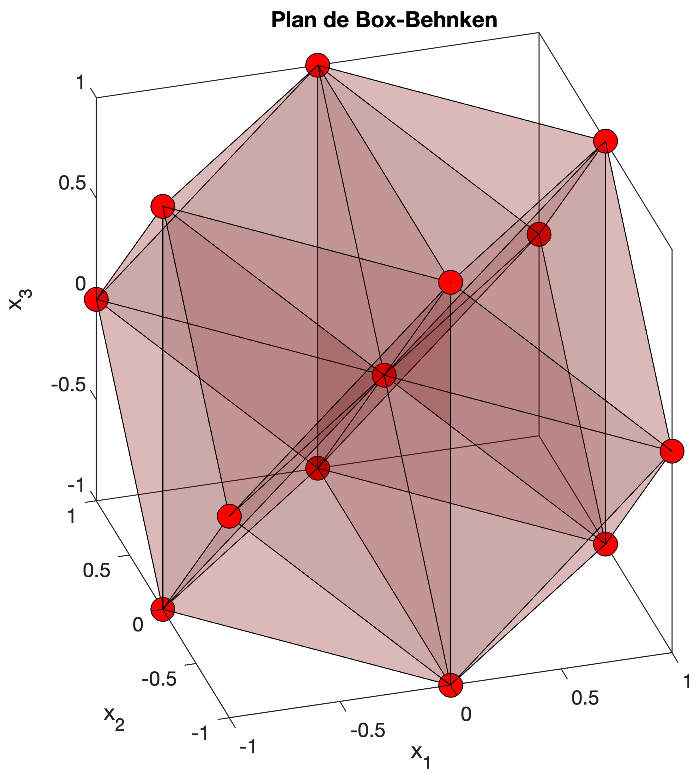
D Box-Behnken

Generating the15 runs of the Box-Behnken design

```
E4= bbdesign(3);
axonometry(E4,'Plan de Box-Behnken');
```

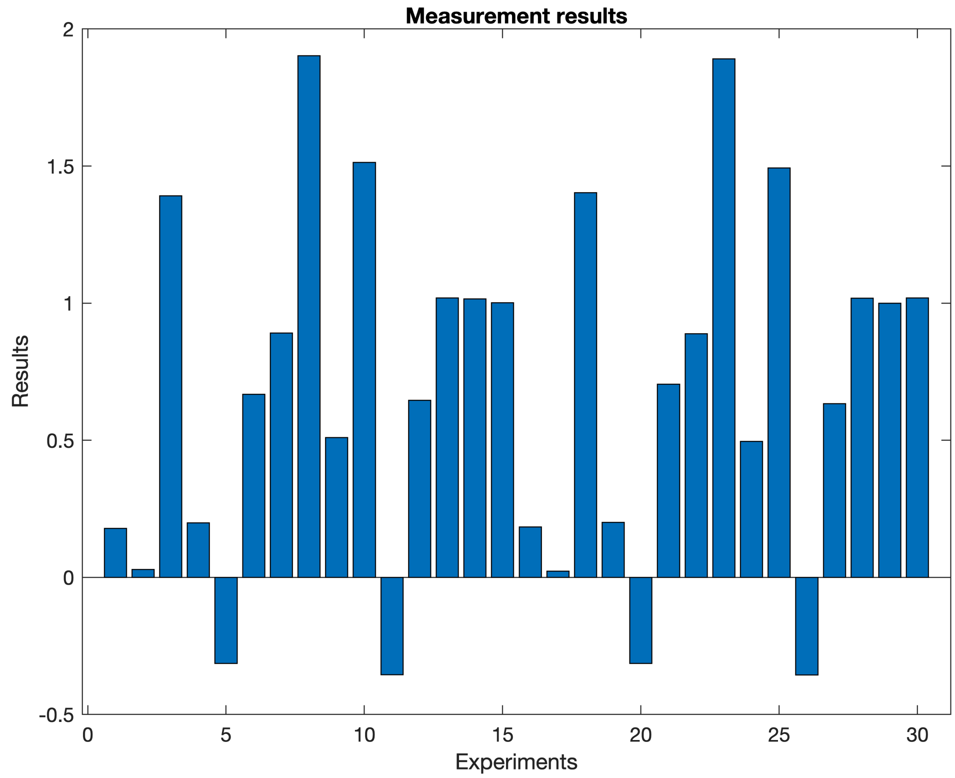
```
Warning: Duplicate data points have been detected and removed.
 The Triangulation indices are defined with respect to the unique set of points in
 delaunayTriangulation.
```

```
view([-16.70 29.35])
```



D.1 Simulating the measurements

```
Y4=mesure3([E4;E4]); % ltwo replicates
figure
bar(Y4)
title('Measurement results')
xlabel('Experiments')
ylabel('Results')
```



D.2 Infering the quadratic model

```
mdl4=fitlm([E4;E4], Y4,'quadratic')
```

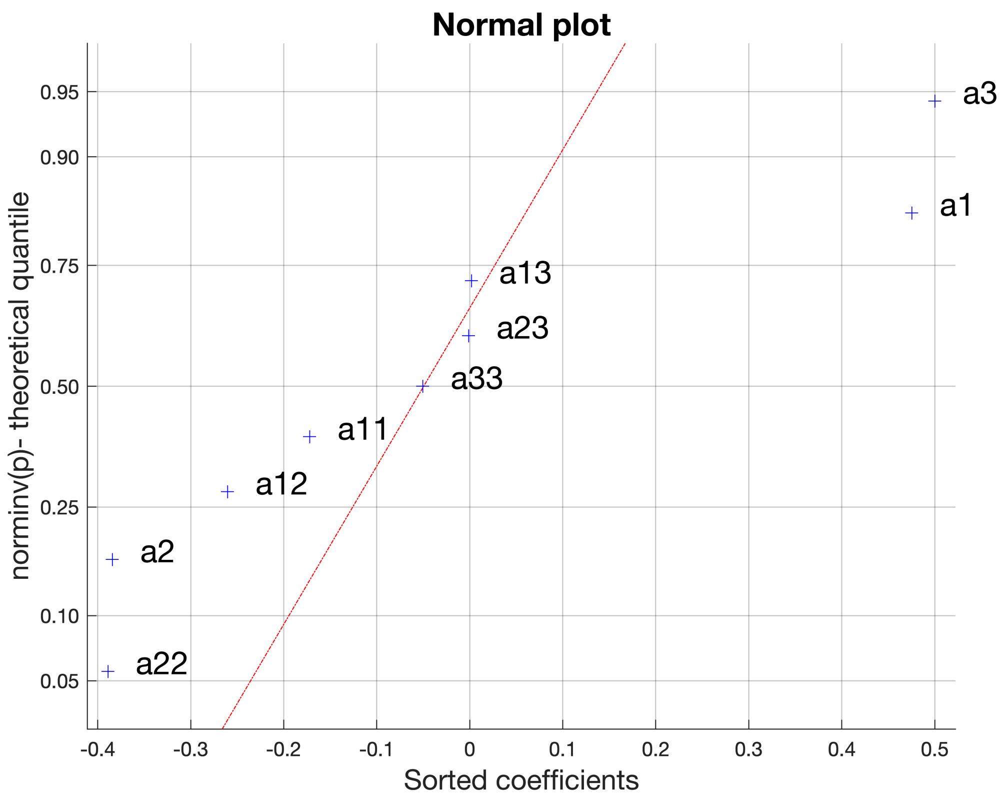
```
mdl4 = 
Linear regression model:
    y ~ 1 + x1*x2 + x1*x3 + x2*x3 + x1^2 + x2^2 + x3^2

Estimated Coefficients:
```

	Estimate	SE	tStat	pValue
(Intercept)	1.0122	0.049863	20.3	8.1294e-15
x1	0.47546	0.030535	15.571	1.2055e-12
x2	-0.38433	0.030535	-12.587	5.8233e-11
x3	0.50023	0.030535	16.382	4.6861e-13
x1:x2	-0.26041	0.043182	-6.0304	6.7762e-06
x1:x3	0.0018257	0.043182	0.042278	0.9667
x2:x3	-0.0012097	0.043182	-0.028014	0.97793
x1^2	-0.17203	0.044946	-3.8275	0.0010526
x2^2	-0.38907	0.044946	-8.6564	3.3758e-08
x3^2	-0.050435	0.044946	-1.1221	0.2751

```
Number of observations: 30, Error degrees of freedom: 20
Root Mean Squared Error: 0.122
R-squared: 0.975,  Adjusted R-Squared: 0.964
F-statistic vs. constant model: 87.9, p-value = 4.22e-14
```

The ad-hoc normplot routine allows to discriminate between retained coefficient to draw an indicavive line:

```
normplot_DOE(mdl4.Coefficients.Estimate(2:end),5:7,-0.4, 0.4,0.03,0.05)
```



The retained coefficients are therefore a_1, a_2, a_3, a_{12} and a_{22} .

Let's define the coefficient matrix of the chosen model

```
spec=[0 0 0
     1 0 0
     0 1 0
     0 0 1
     1 1 0
     2 0 0
     0 2 0];
```

The model is then $\hat{y} = a_o + a_1x_1 + a_2x_2 + a_3x_3 + a_{12}x_1x_2 + a_{11}x_1^2 + a_{22}x_2^2$

Let's infere the chosen model with the *fitlm()* routine

```
mdl_BB=fitlm([E4;E4],Y4,spec)
```

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

Estimated Coefficients:
                   Estimate       SE         tStat       pValue
```

```
                 ________    ________    ________    __________

    (Intercept)     0.98116     0.03989      24.597    5.0951e-18
    x1              0.47546    0.029358      16.195    4.5417e-14
    x2             -0.38433    0.029358     -13.091    3.8189e-12
    x3              0.50023    0.029358      17.039    1.5401e-14
    x1:x2          -0.26041    0.041519     -6.2721    2.1222e-06
    x1^2           -0.16815    0.043086     -3.9027    0.00071607
    x2^2           -0.38519    0.043086       -8.94    6.0587e-09


Number of observations: 30, Error degrees of freedom: 23
Root Mean Squared Error: 0.117
R-squared: 0.974,  Adjusted R-Squared: 0.967
F-statistic vs. constant model: 142, p-value = 5.26e-17
```

```
anova(mdl_BB,'summary',2)
```

ans = 7×5 table

	SumSq	DF	MeanSq	F	pValue
1 Total	12.0935	29	0.4170	NaN	NaN
2 Model	11.7763	6	1.9627	142.3255	5.2559e-17
3 . Linear	9.9839	3	3.3280	241.3272	1.6111e-17
4 . Nonlinear	1.7923	3	0.5974	43.3238	1.2621e-09
5 Residual	0.3172	23	0.0138	NaN	NaN
6 . Lack of fit	0.3156	6	0.0526	563.6510	1.3691e-18
7 . Pure error	0.0016	17	0.0001	NaN	NaN

By inspecting the output of the routine, we observe that all the p-values �� are sufficiently small. Nevertheless the term a_{11} which has a p-value of 0.13%.

The analysis of variance makes it possible to detect a lack of fit, the model deserves to be completed.

D.3 Canonical analysis

The model does not include any cross or cubic term involving the factor x_3, so we can perform the canonical analysis only on x_1 and x_2 , fixing $x_3 = 0$.

Calculating the vector of linear coefficients and the matrix of coefficients of the second degree:

```
coef=mdl_BB.Coefficients.Estimate;
ao=coef(1);
a= coef(2:3);
A= [coef(6) coef(5)/2;coef(5)/2 coef(7)];
```

Computing the fixed point and the eigenvectors:

```
xs= -inv(A)*a/2 % fixed point
```

xs = 2×1

```
    2.4383
   -1.3231
```

```
ys=ao+a'*xs+xs'*A*xs % value of the function at the fixed point
```

```
ys = 1.8151
```

```
[V,L]=eig(A) % Eigenvectors and eigenvalues
```

```
V = 2×2
    0.4241   -0.9056
    0.9056    0.4241
L = 2×2
   -0.4462         0
         0   -0.1072
```

```
teta=atan(V(2,1)/V(1,1))/pi*180 % Angle between the two frames
```

```
teta = 64.9046
```

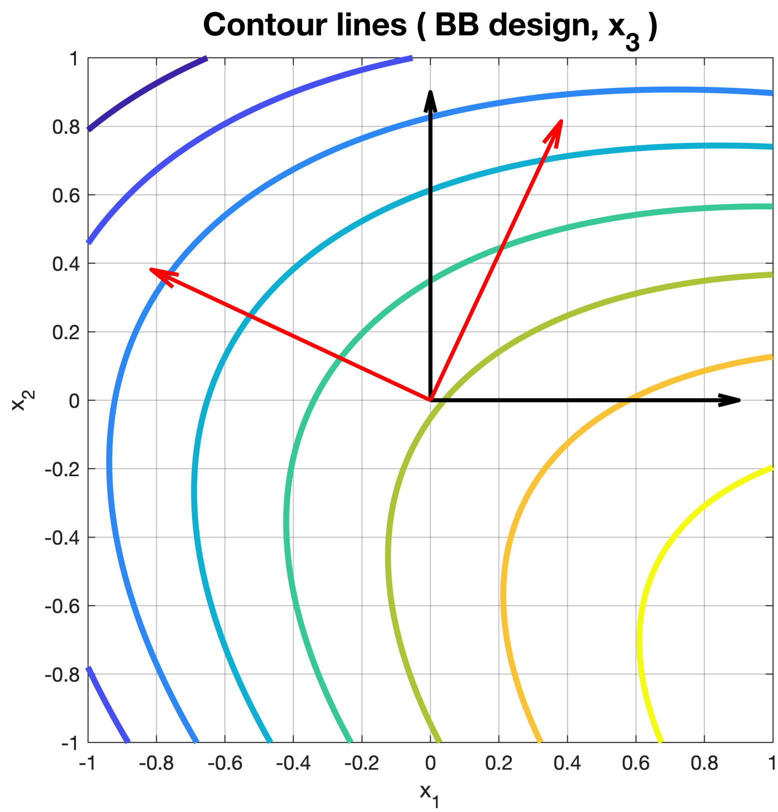
The two eigenvalues of the same sign reveal an elliptical surface. This function decreases as one moves away from the fixed point $(2.67, -1.38)$. The canonical form is

$$Y = a_o + \sum \lambda_i \widetilde{X}_i = 1.88 - 0.45\widetilde{X}_1^2 - 0.10\widetilde{X}_2^2$$

The axes are $\widetilde{X}_1 = 0.56x_1 + 0.83x_2$ and $\widetilde{X}_2 = -0.83x_1 + 0.56x_2$. The two frames have an angle of 56° .

Let's draw the contour lines in the experimental domain

```
syms x1 x2
y4=ao+ a(1)*x1+a(2)*x2+ 2*A(1,2)*x1*x2+ A(1,1)*x1^2+ A(2,2)*x2^2;
figure
fcontour(y4,[-1 1 -1 1],'LineWidth',3)
title('Contour lines ( BB design, x_3 )', 'FontSize',16)
xlabel('x_1')
ylabel('x_2')
pbaspect([1 1 1])
grid on
hold on
quiver([0,0],[0,0],[1 0],[0,1],'k-','LineWidth',2)
quiver([0,0],[0,0],V(1,:),V(2,:),'r-','LineWidth',2)
hold off
```

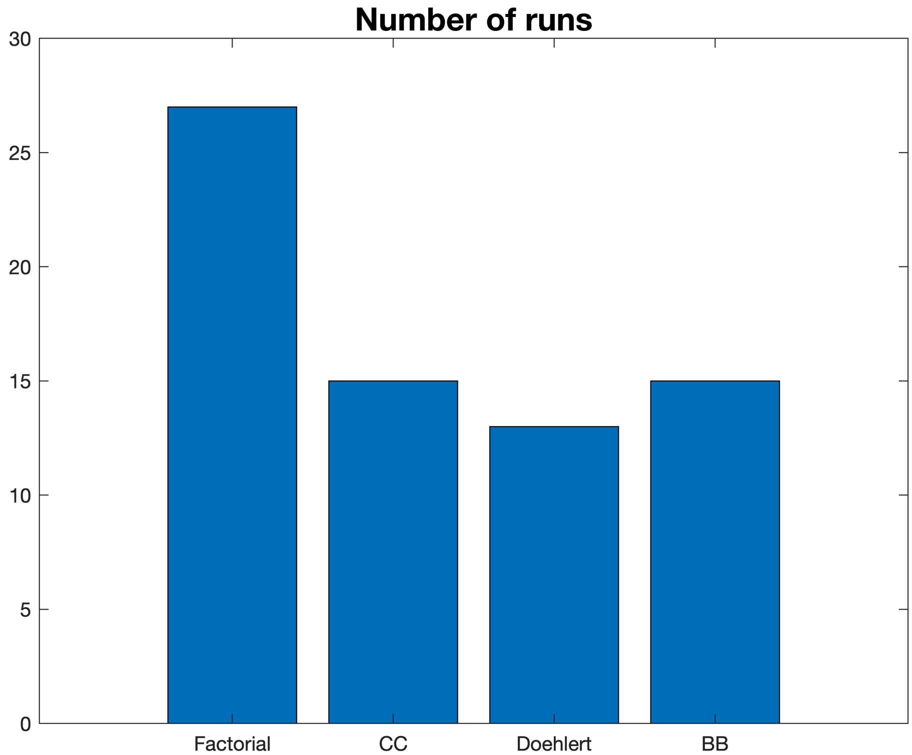


4. Comparison between the different designs

Comparison of the design in terms of number of runs

The most expensive design is the 3^3 factorial design with 27 runs. The most economic design is the Doehlert design with 13 runs

```
bar([27 15 13 15])
title('Number of runs', 'FontSize',16)
set(gca,'XTickLabel',{'Factorial' 'CC' 'Doehlert' 'BB'})
```

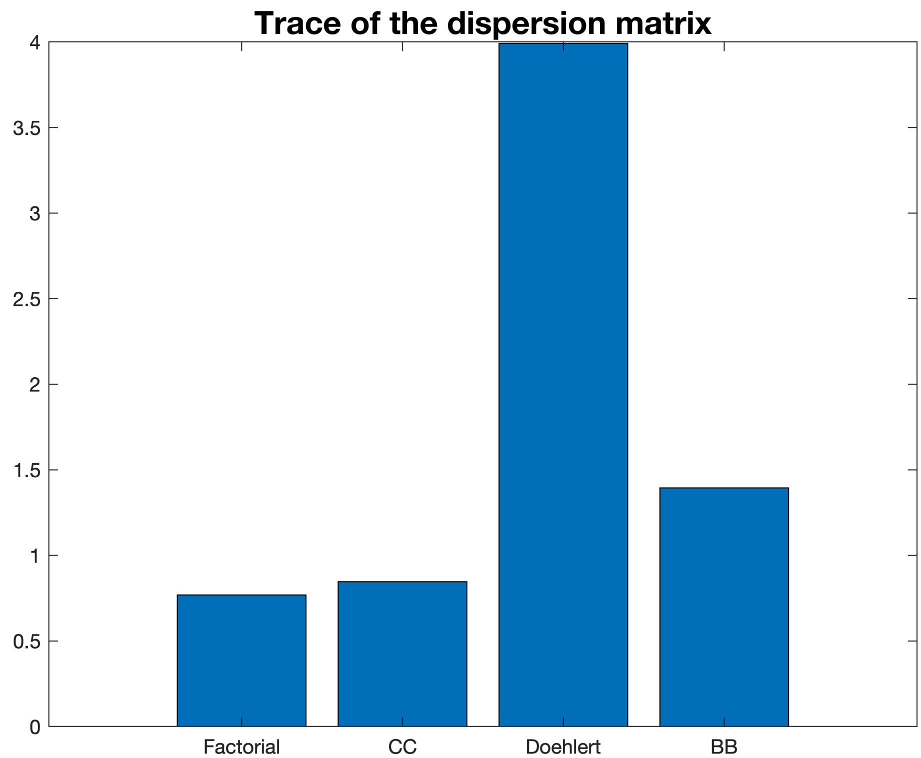


Comparison in term of accuracy

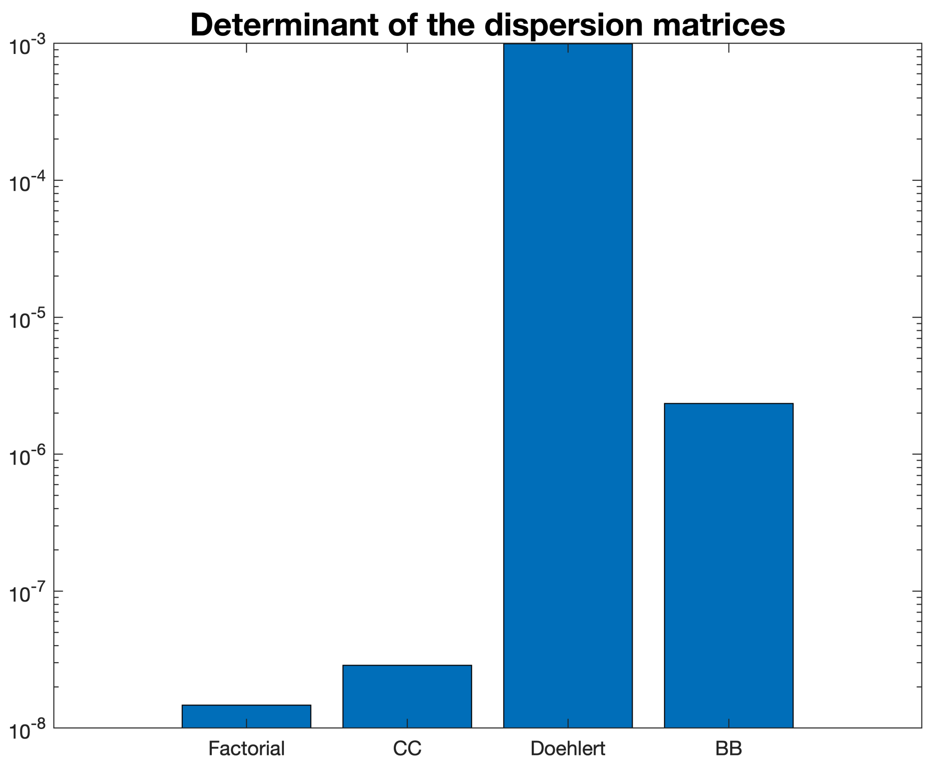
The first point of comparison for the accuracy of the designs is the trace of the dispersion matrix. Not surprisingly, the best design is the design with the maximum of runs, the 3^3 factorial design and the less precise is the Doehlert design. Notice the intermediary position of the BB design

```
mod_1=x2fx(E1,spec);
DD=inv(mod_1'*mod_1);
T(1)=trace(DD);
D(1)=det(DD);

mod_2=x2fx(E2,spec);
DD=inv(mod_2'*mod_2);
T(2)=trace(DD);
D(2)=det(DD);

mod_3=x2fx(E3,spec);
DD=inv(mod_3'*mod_3);
T(3)=trace(DD);
D(3)=det(DD);

mod_4=x2fx(E4,spec);
DD=inv(mod_4'*mod_4);
T(4)=trace(DD);
D(4)=det(DD);

bar(T)
```

```
title('Trace of the dispersion matrix', 'FontSize',16)
set(gca,'XTickLabel',{'Factorial' 'CC' 'Doehlert' 'BB'})
```




```
bar(D)
title('Determinant of the dispersion matrices', 'FontSize',16)
set(gca,'XTickLabel',{'Factorial' 'CC' 'Doehlert' 'BB'},'YScale','log')
```



At the level of the determinant of the dispersion matrix, the figure is the same.

Conclusion

- The 3^3 factorial design is the most expensive one with a difference of 10 runs more than the other ones (the 20 when the design is duplicated),
- For the trace (which is propostional to the SS of the confidence intervals) it is possible to observe that the Dohlert design is at leaste two times less precise than the other ones,
- For the ratio quality/runs the CC design is the most interesting.

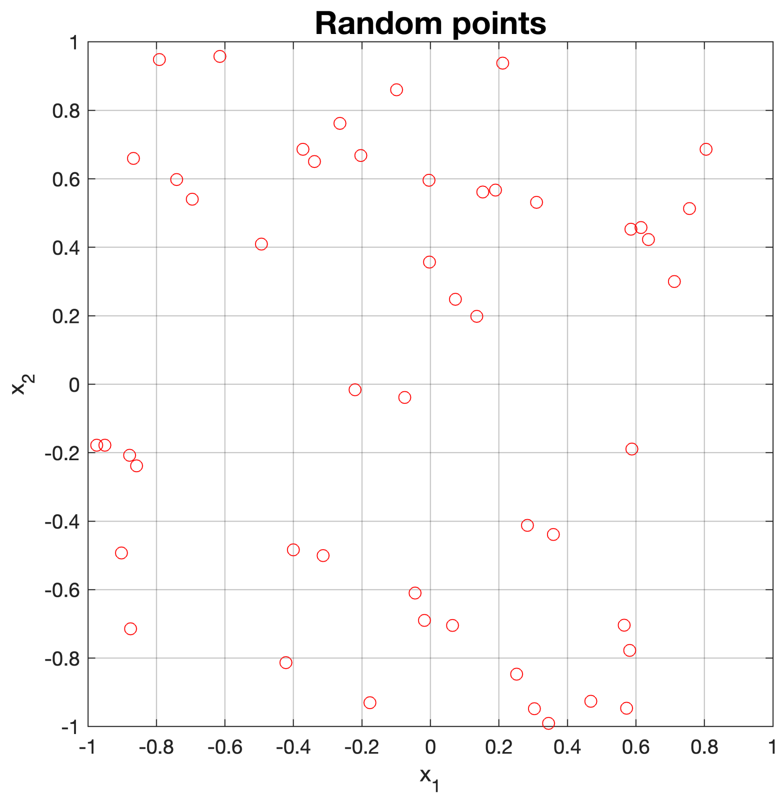
5. Comparison of 50 random points

The number of points has been increased in comparison with the statement of the exercise to better put in evidence the tendency

Selecting random points in the plane $x_3 = 0$:

```
E_alea=(rand(50,3)-.5)*2;

figure
plot(E_alea(:,1),E_alea(:,2),'or')
title('Random points','FontSize',16)
xlabel('x_1')
ylabel('x_2')
box on
```

```
grid on
pbaspect([1 1 1])
```



The measurement is simulated with the routine *mesure3():*

```
Y_alea=mesure3(E_alea);
```

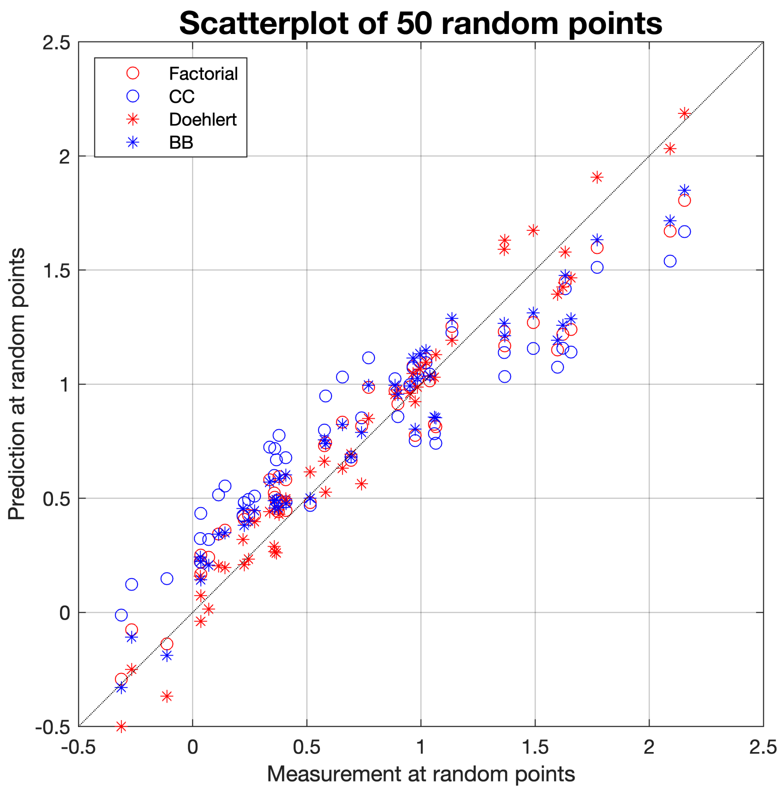
The corresponding predictions for the four fitted model is done with the routine *predict():*

```
[YP1,YciP1]=predict(mdl_fact,E_alea);
[YP2,YciP2]=predict(mdl_CC,E_alea);
[YP3,YciP3]=predict(mdl_doe,E_alea);
[YP4,YciP4]=predict(mdl_BB,E_alea);
```

A comparison is done between the measurement and the different predictions:

```
figure
plot(Y_alea,YP1,'or')
hold on
plot(Y_alea,YP2,'ob')
plot(Y_alea,YP3,'*r')
plot(Y_alea,YP4,'*b')
plot([-0.5,2.5],[-0.5,2.5],':k')
hold off
grid on
pbaspect([1 1 1])
legend({'Factorial' 'CC' 'Doehlert' 'BB'},'location', 'northwest')
title('Scatterplot of 50 random points','FontSize',16)
```

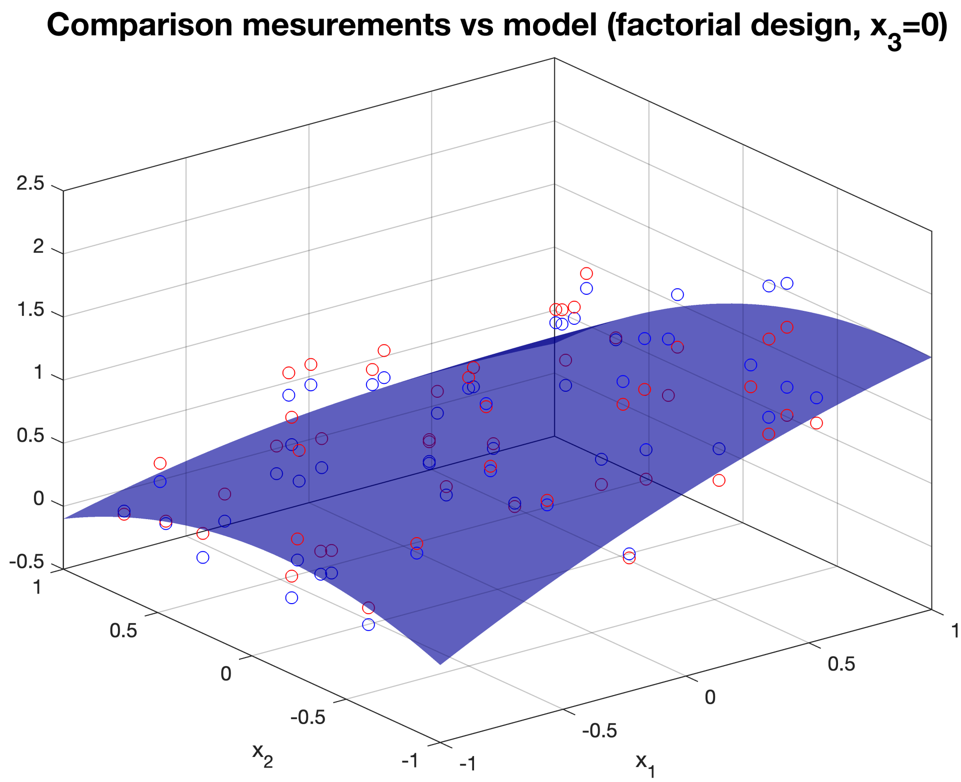
```
xlabel('Measurement at random points')
ylabel('Prediction at random points')
```



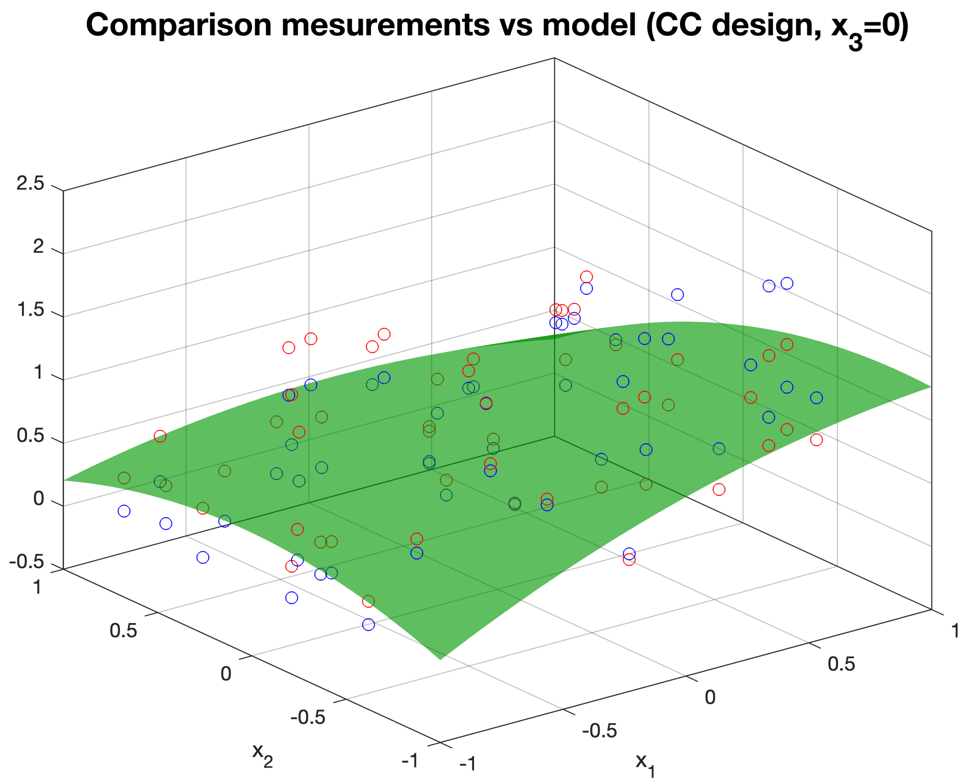
If we make successive simulations, we can observe that the pattern is reproducible.

Let's compare the randomly measured points to the different response surfaces

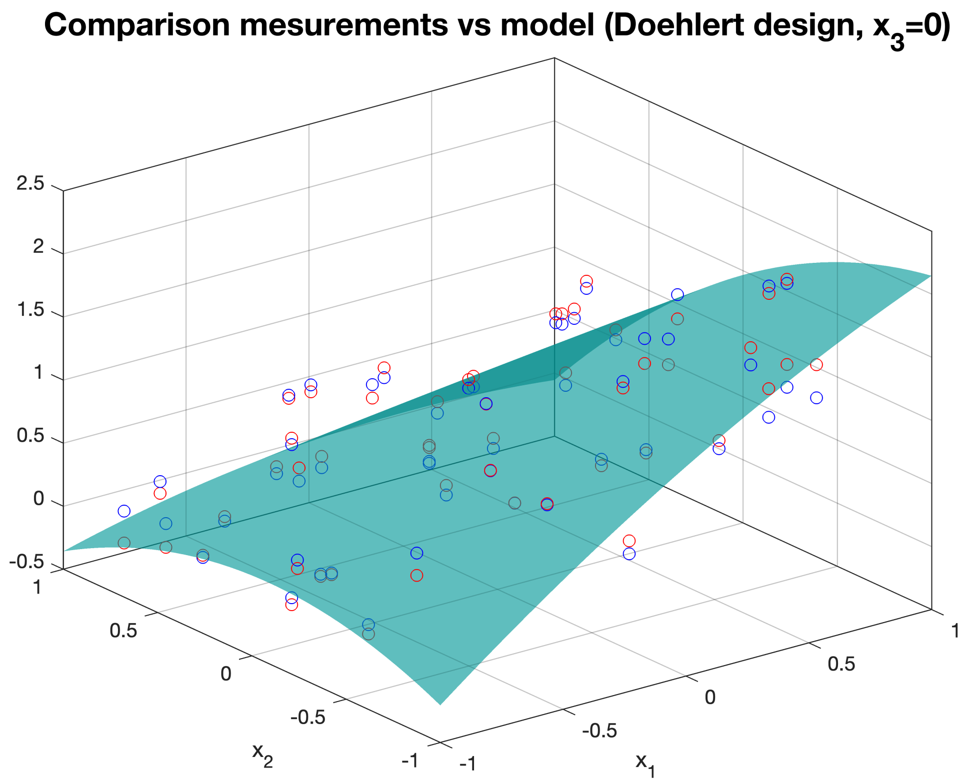
```
% model 1 (factorial design)
figure
plot3(E_alea(:,1),E_alea(:,2),Y_alea,'ob')
hold on
plot3(E_alea(:,1),E_alea(:,2),YP1,'or')
fsurf(y1,[-1 1 -1 1],'FaceAlpha',0.5,'FaceColor','blue','EdgeColor','none')
hold off
box on
grid on
title('Comparison mesurements vs model (factorial design, x_3=0)','FontSize',16)
xlabel('x_1')
ylabel('x_2')
```



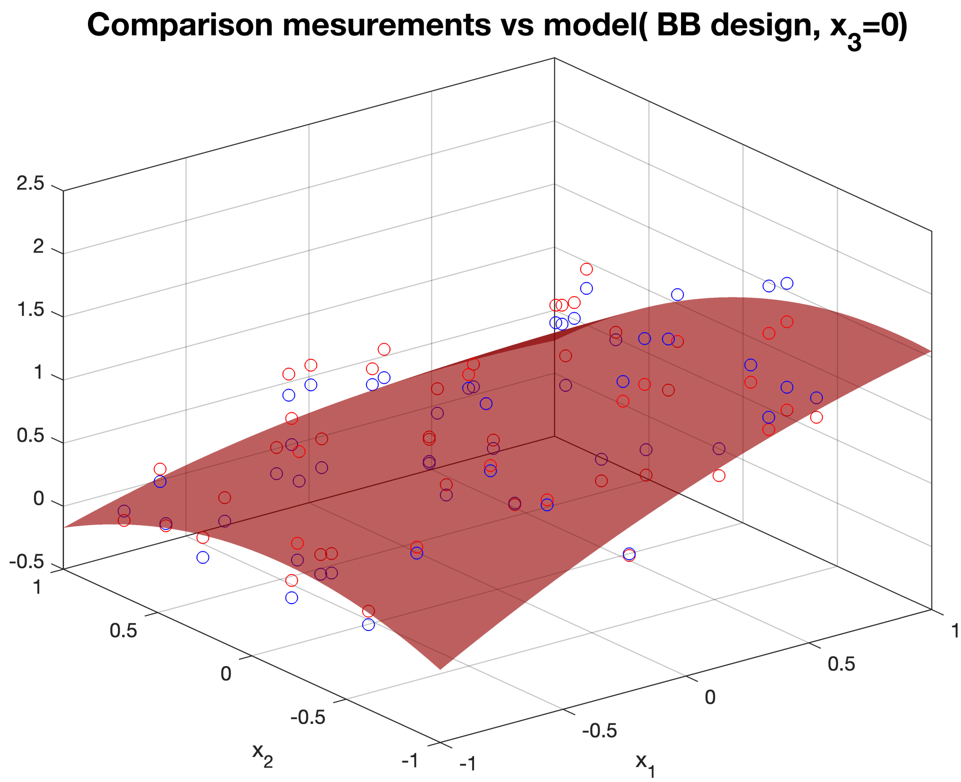

```
% model 2 (CC design)
figure
plot3(E_alea(:,1),E_alea(:,2),Y_alea,'ob')
hold on
plot3(E_alea(:,1),E_alea(:,2),YP2,'or')
fsurf(y2,[-1 1 -1 1],'FaceAlpha',0.5,'FaceColor','green','EdgeColor','none')
hold off
box on
grid on
title('Comparison mesurements vs model (CC design, x_3=0)','FontSize',16)
xlabel('x_1')
ylabel('x_2')
```




```
% model 3 (Doehlert design)
figure
plot3(E_alea(:,1),E_alea(:,2),Y_alea,'ob')
hold on
plot3(E_alea(:,1),E_alea(:,2),YP3,'or')
fsurf(y3,[-1 1 -1 1],'FaceAlpha',0.5,'FaceColor','cyan','EdgeColor','none')
hold off
box on
grid on
title('Comparison mesurements vs model (Doehlert design, x_3=0)','FontSize',16)
xlabel('x_1')
ylabel('x_2')
```



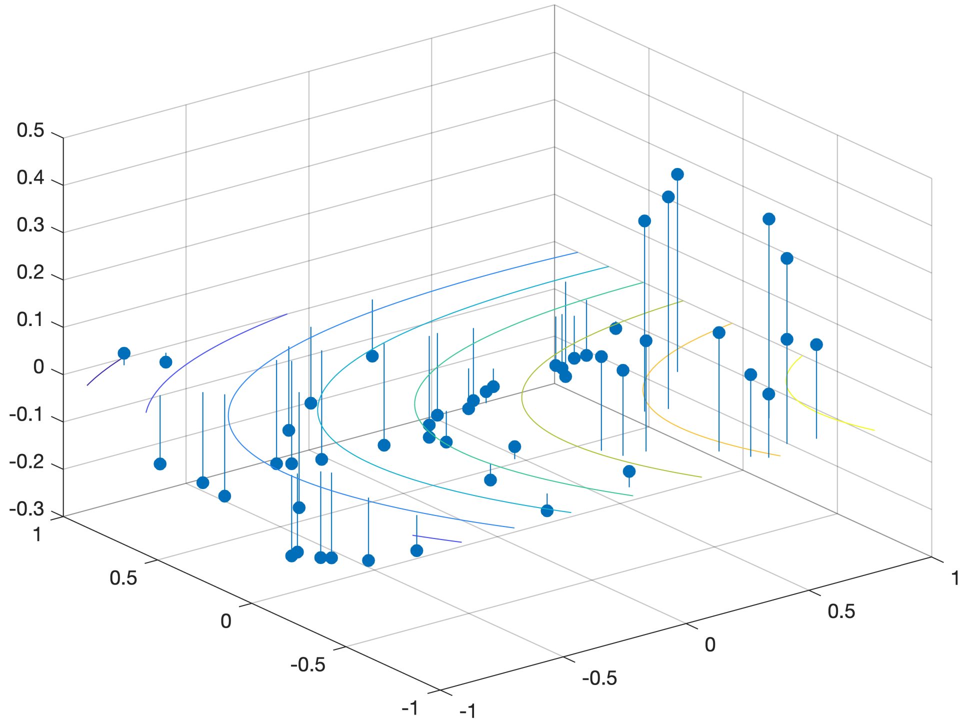

```
% model 4 (Box Behnken)
figure
plot3(E_alea(:,1),E_alea(:,2),Y_alea,'ob')
hold on
plot3(E_alea(:,1),E_alea(:,2),YP4,'or')
fsurf(y4,[-1 1 -1 1],'FaceAlpha',0.5,'FaceColor','red','EdgeColor','none')
hold off
box on
grid on
title('Comparison mesurements vs model( BB design, x_3=0)','FontSize',16)
xlabel('x_1')
ylabel('x_2')
```



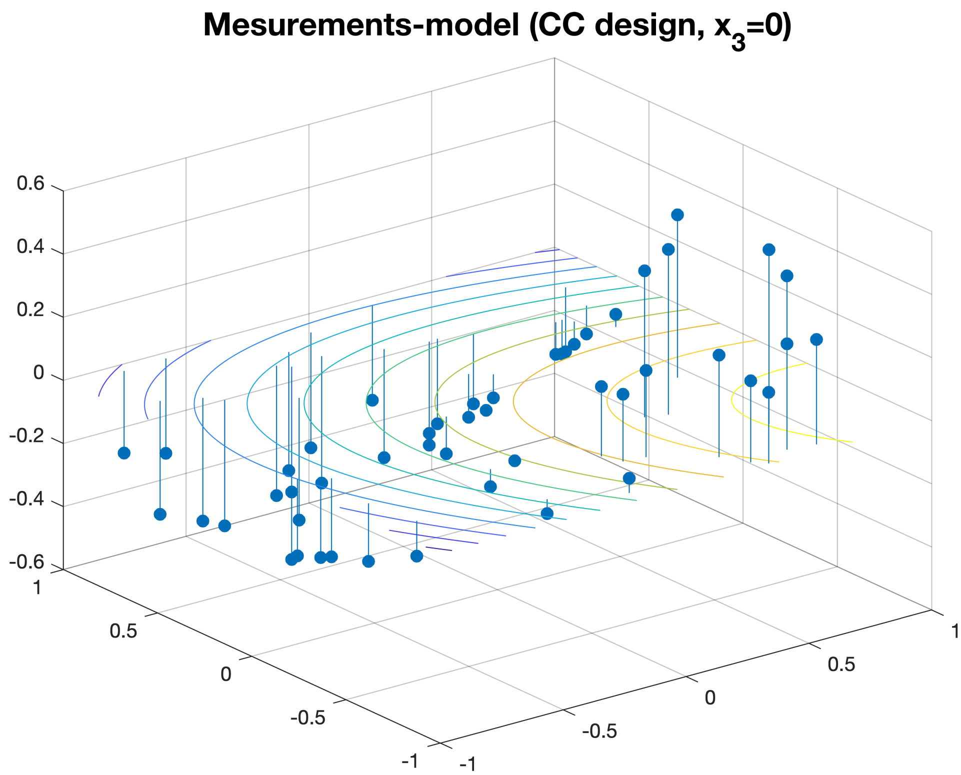
Another visualization of the model-measurement difference from a residual (this is also something that can be done at the level of the residuals when fitting the model)

```
% modèle 1
figure
stem3(E_alea(:,1),E_alea(:,2),Y_alea-YP1,'filled')
hold on
fcontour(y1,[-1 1 -1 1])
hold off
title('Mesurements-model (Factorial design, x_3=0)','FontSize',16)
```

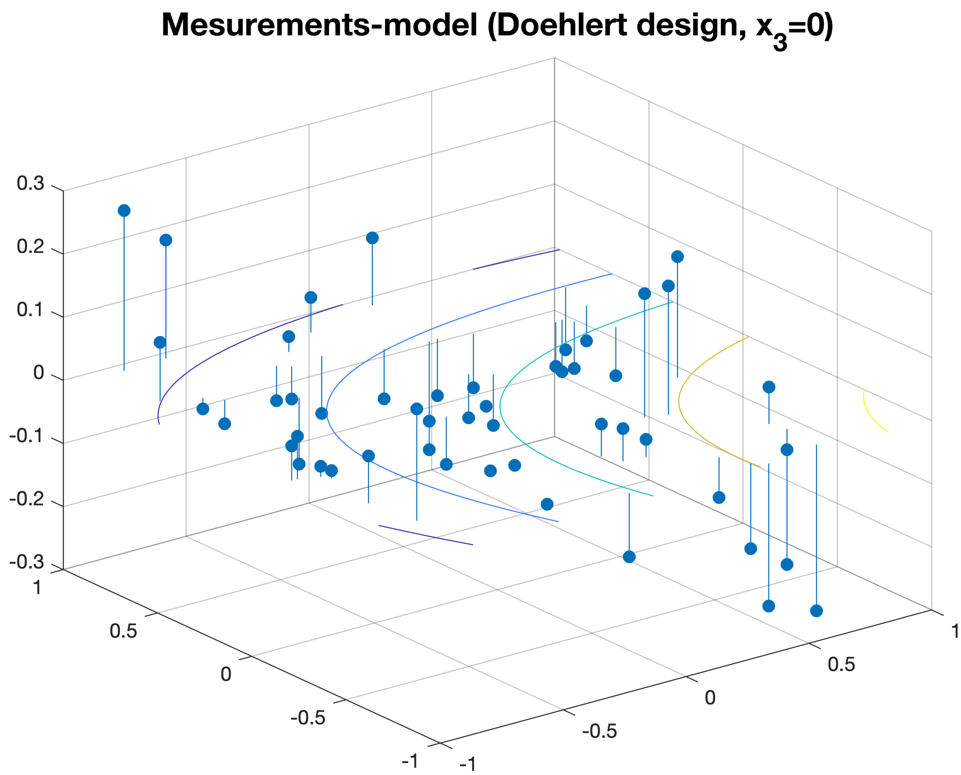
Mesurements-model (Factorial design, x_3=0)



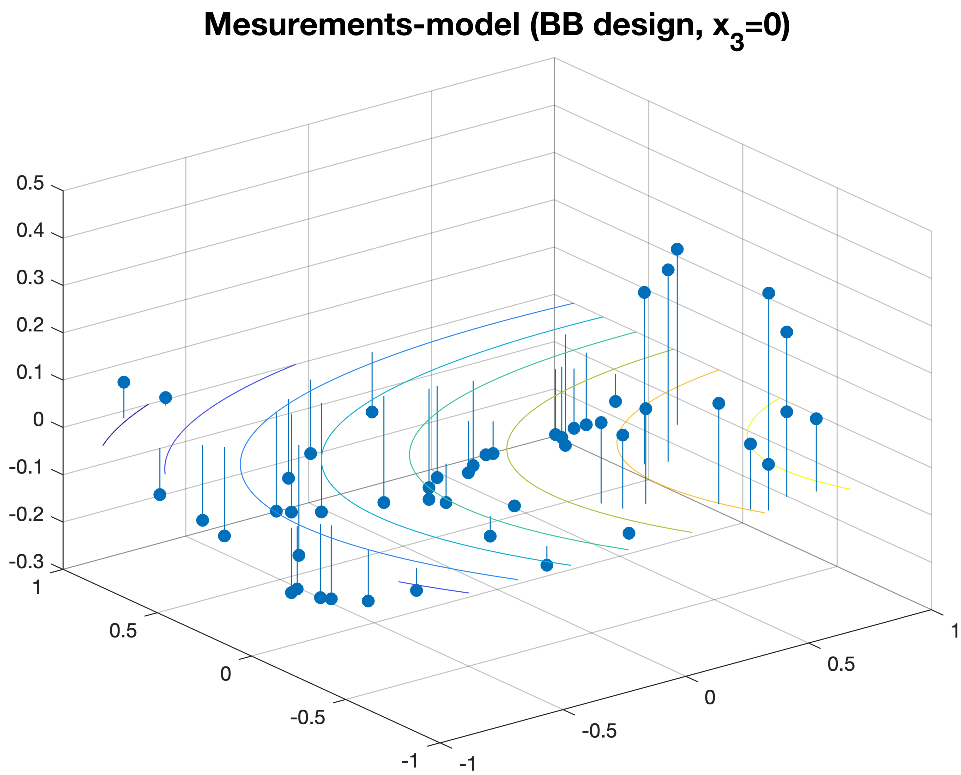

```
% Modèle 2
figure
stem3(E_alea(:,1),E_alea(:,2),Y_alea-YP2,'filled')
hold on
fcontour(y2,[-1 1 -1 1])
hold off
title('Mesurements-model (CC design, x_3=0)','FontSize',16)
```




```
% Modèle 3
figure
stem3(E_alea(:,1),E_alea(:,2),Y_alea-YP3,'filled')
hold on
fcontour(y3,[-1 1 -1 1])
hold off
title('Mesurements-model (Doehlert design, x_3=0)','FontSize',16)
```




```
% Modèle 4
figure
stem3(E_alea(:,1),E_alea(:,2),Y_alea-YP4,'filled')
hold on
fcontour(y4,[-1 1 -1 1])
hold off
title('Mesurements-model (BB design, x_3=0)','FontSize',16)
```



At the level of residuals for random points, the model obtained with a Doehlert design is the most favorable. You have to realize that it is rather a coincidence. All the regressions have significant lack of fit, so they all have a visible defect compared to the real system represented by the measurement. It turns out that ultimately "the least wrong" is the one obtained by Doehlert's plan. But of course that's not a rule.

6. Visualization with the function *slice*

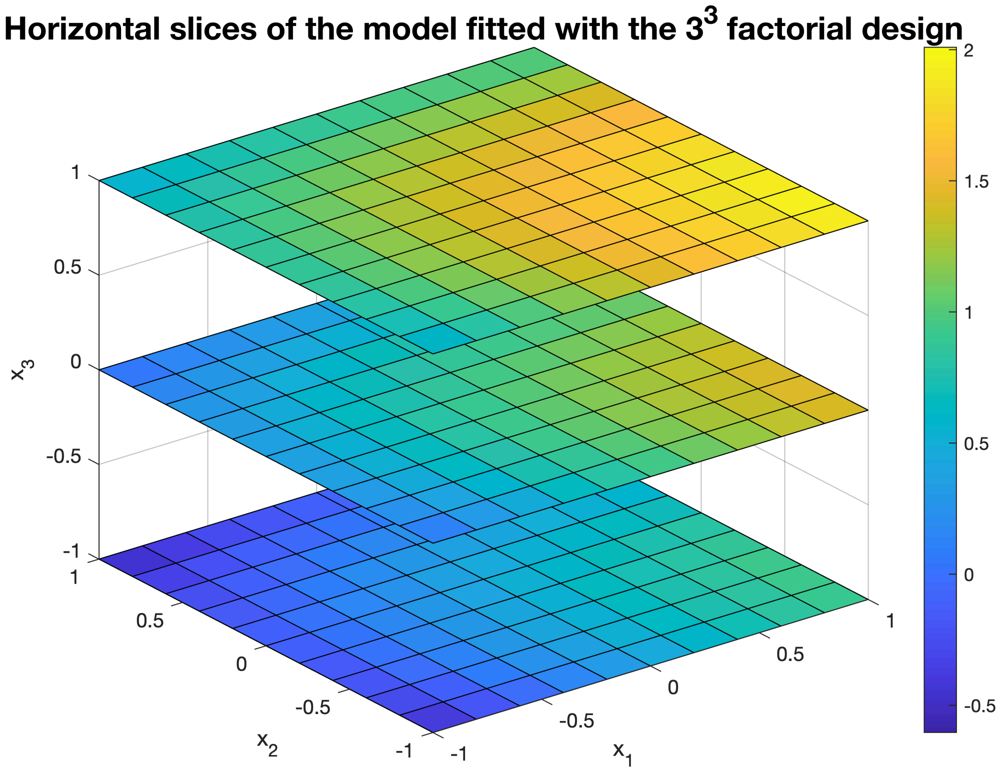
- The slice function allows to represent by a colored plane the values ��that a three-dimensional function takes in a given plane
- For this illustration we will consider the model obtained with the first plane (the factorial plane 3^3)
- We start by calculating a block of data corresponding to the parallelepipedic domain (-1,1)

```
[XX1,XX2,XX3]=meshgrid(-1:.2:1); % generating the mesh on which the model is computed
coef=mdl_fact.Coefficients.Estimate;
YY=coef(1) + coef(2)* XX1+ coef(3)*XX2 + coef(4)*XX3 + coef(5)*XX1.*XX2 +...
    coef(6)* XX1.^2 + coef(7) * XX2.^2; % evaluating the function
```

Then it is possible to visualize horizontal planes at three levels $x_3 = -1, 0, +1$

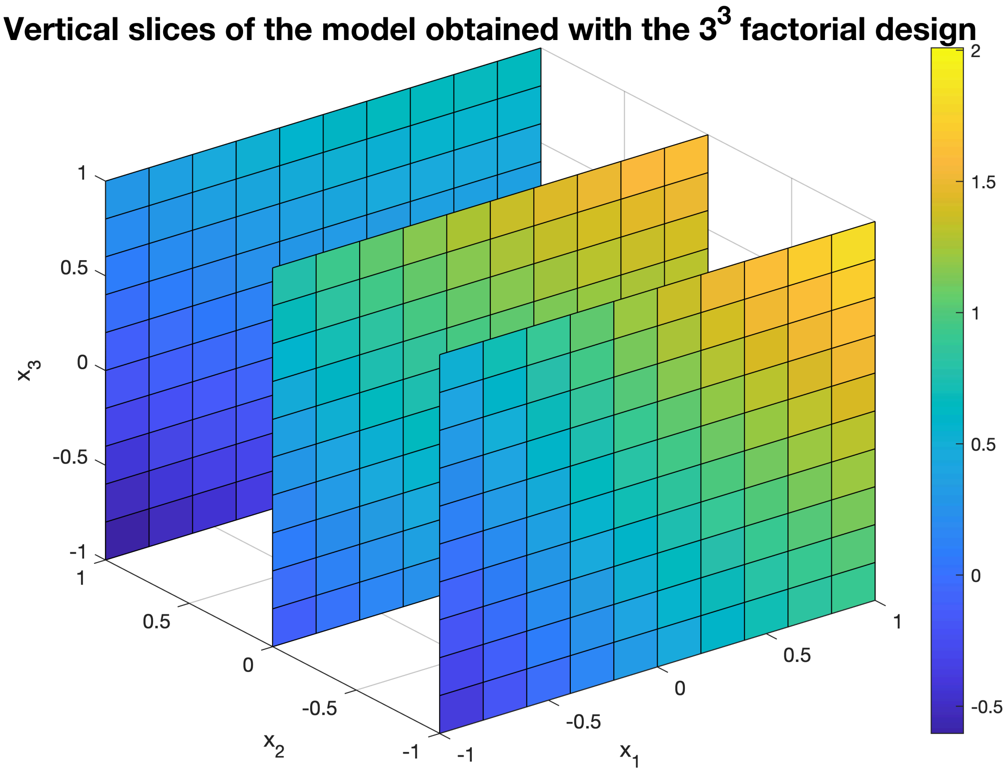
```
figure
xslice=[];
yslice=[];
```

```
zslice=[-1,0,1];
slice(XX1,XX2,XX3,YY,xslice,yslice,zslice)
colorbar
title('Horizontal slices of the model fitted with the 3^3 factorial design','FontSize',
xlabel('x_1')
ylabel('x_2')
zlabel('x_3')
```



Or vertical planes $x_2 = -1, 0, +1$

```
figure
xslice=[];
zslice=[];
yslice=[-1,0,1];
slice(XX1,XX2,XX3,YY,xslice,yslice,zslice)
colorbar
title('Vertical slices of the model obtained with the 3^3 factorial design','FontSize',
xlabel('x_1')
ylabel('x_2')
zlabel('x_3')
```



7. Visualization with isosurfaces

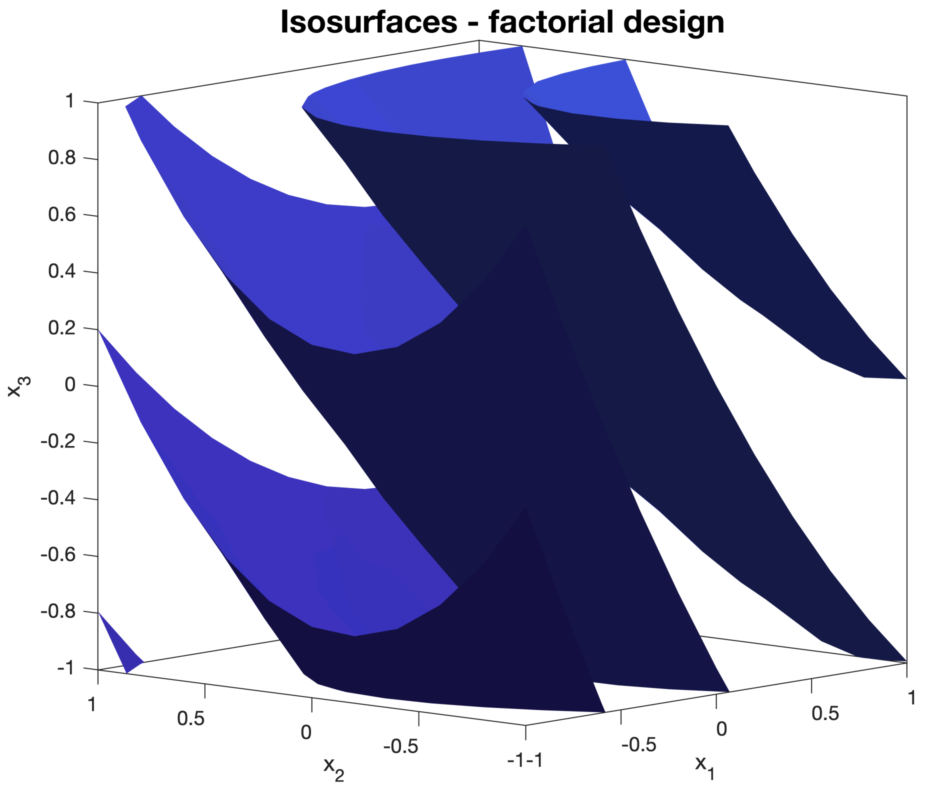
Keep in memory the colormap

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

Drawing of the isosurface

```
figure
valeur= -.5:.5:2;
for k=1:5
    p=patch(isosurface(XX1,XX2,XX3,YY,valeur(k)));
    isonormals(XX1,XX2,XX3,YY,p)
    p.FaceColor=map(10+k*7,1:3);
    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('Isosurfaces - factorial design','FontSize',16)
xlabel('x_1')
ylabel('x_2')
zlabel('x_3')
box on
```



Functions

F1: Axonometry function

Function used to draw a 3D representation of the design *E*

```
function axonometry(E,libele_titre)

figure
plot3(E(:,1),E(:,2),E(:,3),'ok',...
    'MarkerFaceColor','red',...
    'MarkerSize',12)
hold on
T=delaunayTriangulation(E);
tetramesh(T,'FaceColor','red','FaceAlpha',0.05)
hold off
```

```
box on
pbaspect([1 1 1])
title(libele_titre)
xlabel('x_1')
ylabel('x_2')
zlabel('x_3')

view([-59.90 30.95])
end
```

F2: Selective Normal plot function

Function drawing a normal plot discriminating between the significant and non significant coefficients

```
function normplot_DOE(coefficients,coef_non_significant,xmin, xmax,dx,dy)
% coefficients: coefficients of the polinomial model (without the constant)
% coef_non_significatif:interval of indices corresponding to the neglected coefficients
%
% xmin: minimal abscissa of the right line
% xmax: maximal abscissa of the right line
% dx: xshift for the labels
% dy: yshift for the lables



% labels of the coefficients for a quadratic model of three fgactors
label={'a1' 'a2' 'a3' 'a12' 'a13' 'a23' 'a11' 'a22'  'a33' };


% Draw the Normal plot
figure
h=normplot(coefficients);
title('Normal plot','FontWeight','bold','FontSize',16)
xlabel('Sorted coefficients','FontSize',14)
ylabel('norminv(p)- theoretical quantile','FontSize',14)

%Re-compute the indicative right line
x=get(h(1),'Xdata')';
x=[ones(size(coef_non_significant,2),1),x(coef_non_significant)];
y=get(h(1),'Ydata')';
y=y(coef_non_significant);
alpha=inv(x'*x)*x'*y;
delete(h(2))
set(h(3),'Xdata',[xmin,xmax],'Ydata', [1 xmin;1 xmax]*alpha)

% Write the labels
```

```
[C,index]=sort(coefficients);
text(h(1).XData+dx, h(1).YData+dy, label(index),'FontSize',16)

end
```