Section 1

MANE 6313

Subsection 1

Week 14, Module A

Student Learning Outcome

- Select an appropriate experimental design with one or more factors,
- Select an appropriate model with one or more factors,
- Evaluate statistical analyses of experimental designs,
- Assess the model adequacy of any experimental design, and
- Interpret model results.

Module Learning Outcome

Evaluate central composite designs using R

Central Composite Design in R

rsm (version 2.10.3)

ccd: Generate central-composite designs and associated building blocks

Description

These functions generate central-composite designs, or building blocks thereof. They allow for flexible choices of replications, aliasing of predictors and fractional blocks, and choices of axis or 'star' points.

Usage

```
cube(basis, generators, n0 = 4, reps = 1, coding, randomize = TRUE,
    blockgen, bid = 1, inscribed = FALSE)
star(basis, n0 = 4, alpha = "orthogonal", reps = 1, randomize = TRUE)
dupe(design, randomize = TRUE, coding)
foldover(basis, variables, bid, randomize = TRUE)
ccd(basis, generators, blocks = "Block", n0 = 4, alpha = "orthogonal",
    wbreps = 1, bbreps = 1, randomize = TRUE, inscribed = FALSE,
    coding, oneblock = FALSE)
```

ccd.pick()

ccd.pick: Find a good central-composite design

Description

This function looks at all combinations of specified design parameters for central-composite designs, calculates other quantities such as the `alpha` values for rotatability and orthogonal blocking, imposes specified restrictions, and outputs the best combinations in a specified order. This serves as an aid in identifying good designs. The design itself can then be generated using `ccd`, or in pieces using `cube`, `star`, etc.

Usage

```
ccd.pick(k, n.c = 2^k, n0.c = 1:10, blks.c = 1, n0.s = 1:10, bbr.c = 1,
         wbr.s = 1, bbr.s = 1, best = 10, sortby = c("agreement", "N"),
         restrict)
```

Arguments

k	Number of factors in the design
n.c	Number(s) of factorial points in each cube block
n0.c	Numbers(s) of center points in each cube block
blks.c	Number(s) of cube blocks that together comprise one rep of the cube portion
n0.s	Numbers(s) of center points in each star (axis-point) block
bbr.c	Number(s) of copies of each cube block
wbr.s	Number(s) of replications of each star poit within a block
bbr.s	Number(s) of copies of each star block
best	How many designs to list. Use `best=NULL` to list them all
sortby	String(s) containing numeric expressions that are each evaluated and used as sorting key(s). Specify `sortby=NULL` if no sorting is desired.

Example Problem

Table 4.7 Original and Coded Levels of Independent Variables

Variable	Original	Coded −2	−1	0	1	2
Heating temperature	X_1 (°C/30 min)	65.0	70.0	75.0	80.0	85.0
pH	X_2	4.0	5.0	6.0	7.0	8.0
Redox potential	X_3 (volt)	−0.025	0.075	0.175	0.275	0.375
Sodium oxalate	X_4 (Molar)	0.0	0.0125	0.025	0.0375	0.05
Sodium lauryl sulfate	X_5 (% of solids)	0.0	0.05	0.10	0.15	0.20

Source: S. H. Richert, C. V. Morr, and C. M. Cooney (1974). Reprinted from Journal of Food Science. Copyright © by Institute of Food Technologies.

Data from Example Problem

Table 4.6 Experimental Design (Coded) and Response Values

x_1	x_2	x_3	x_4	x_5	Y(%)
−1	−1	−1	−1	1	80.6
1	−1	−1	−1	−1	67.9
−1	1	−1	−1	−1	83.1
1	1	−1	−1	1	38.1
−1	−1	1	−1	−1	79.7
1	−1	1	−1	1	74.7
−1	1	1	−1	1	71.2
1	1	1	−1	−1	36.8
−1	−1	−1	1	−1	81.7
1	−1	−1	1	1	66.8
−1	1	−1	1	1	73.0
1	1	−1	1	−1	40.5
−1	−1	1	1	1	74.9
1	−1	1	1	−1	74.2
−1	1	1	1	−1	63.5
1	1	1	1	1	42.8
−2	0	0	0	0	80.9
2	0	0	0	0	42.4
0	−2	0	0	0	73.4
0	2	0	0	0	45.0
0	0	−2	0	0	66.0
0	0	2	0	0	71.7
0	0	0	−2	0	77.5
0	0	0	2	0	76.3
0	0	0	0	−2	67.4
0	0	0	0	2	86.5
0	0	0	0	0	77.4
0	0	0	0	0	74.6
0	0	0	0	0	79.8
0	0	0	0	0	78.3
0	0	0	0	0	74.8

Source: S. H. Richert, C. V. Morr, and C. M. Cooney (1974). Reprinted from Journal of Food Science. Copyright © by Institute of

ccd() to Create Data Frame

```{r}
library(rsm)
ccd1.coded <- ccd(4,n0=c(0,6),alpha="rotatable",randomize=FALSE,generators=x5~x1*x2*x3*x4)
print(ccd1.coded)
```

```
   run.order std.order x1.as.is x2.as.is x3.as.is x4.as.is x5.as.is Block
1          1         1       -1       -1       -1       -1        1     1
2          2         2        1       -1       -1       -1       -1     1
3          3         3       -1        1       -1       -1       -1     1
4          4         4        1        1       -1       -1        1     1
5          5         5       -1       -1        1       -1       -1     1
6          6         6        1       -1        1       -1        1     1
7          7         7       -1        1        1       -1        1     1
8          8         8        1        1        1       -1       -1     1
9          9         9       -1       -1       -1        1       -1     1
10        10        10        1       -1       -1        1        1     1
11        11        11       -1        1       -1        1        1     1
12        12        12        1        1       -1        1       -1     1
13        13        13       -1       -1        1        1        1     1
14        14        14        1       -1        1        1       -1     1
15        15        15       -1        1        1        1       -1     1
16        16        16        1        1        1        1        1     1
17         1         1       -2        0        0        0        0     2
18         2         2        2        0        0        0        0     2
19         3         3        0       -2        0        0        0     2
20         4         4        0        2        0        0        0     2
21         5         5        0        0       -2        0        0     2
22         6         6        0        0        2        0        0     2
23         7         7        0        0        0       -2        0     2
24         8         8        0        0        0        2        0     2
25         9         9        0        0        0        0       -2     2
26        10        10        0        0        0        0        2     2
27        11        11        0        0        0        0        0     2
28        12        12        0        0        0        0        0     2
29        13        13        0        0        0        0        0     2
30        14        14        0        0        0        0        0     2
31        15        15        0        0        0        0        0     2
32        16        16        0        0        0        0        0     2

Data are stored in coded form using these coding formulas ...
x1 ~ x1.as.is
x2 ~ x2.as.is
x3 ~ x3.as.is
```

ccd() with Uncoded Data

```{r}
library(rsm)
ccd1.df <- ccd(~x1+x2+x3+x4,n0=c(0,6),alpha="rotatable",randomize=FALSE,coding = list(x1~(Temp-75)/5,x2~(pH-6),x3~(redox-0.175)/0.1,x4~(naOxalate-0.025)/0.0125,x5~(naLauryl-0.1)/0.05),oneblock=TRUE,generators=x5~x1*x2*x3*x4)
print(ccd1.df)
```

```
   run.order std.order Temp pH  redox naOxalate naLauryl
1          1         1   70  5  0.075    0.0125     0.15
2          2         2   80  5  0.075    0.0125     0.05
3          3         3   70  7  0.075    0.0125     0.05
4          4         4   80  7  0.075    0.0125     0.15
5          5         5   70  5  0.275    0.0125     0.05
6          6         6   80  5  0.275    0.0125     0.15
7          7         7   70  7  0.275    0.0125     0.15
8          8         8   80  7  0.275    0.0125     0.05
9          9         9   70  5  0.075    0.0375     0.05
10        10        10   80  5  0.075    0.0375     0.15
11        11        11   70  7  0.075    0.0375     0.15
12        12        12   80  7  0.075    0.0375     0.05
13        13        13   70  5  0.275    0.0375     0.15
14        14        14   80  5  0.275    0.0375     0.05
15        15        15   70  7  0.275    0.0375     0.05
16        16        16   80  7  0.275    0.0375     0.15
17         1         1   65  6  0.175    0.0250     0.10
18         2         2   85  6  0.175    0.0250     0.10
19         3         3   75  4  0.175    0.0250     0.10
20         4         4   75  8  0.175    0.0250     0.10
21         5         5   75  6 -0.025    0.0250     0.10
22         6         6   75  6  0.375    0.0250     0.10
23         7         7   75  6  0.175    0.0000     0.10
24         8         8   75  6  0.175    0.0500     0.10
25         9         9   75  6  0.175    0.0250     0.00
26        10        10   75  6  0.175    0.0250     0.20
27        11        11   75  6  0.175    0.0250     0.10
28        12        12   75  6  0.175    0.0250     0.10
29        13        13   75  6  0.175    0.0250     0.10
30        14        14   75  6  0.175    0.0250     0.10
31        15        15   75  6  0.175    0.0250     0.10
32        16        16   75  6  0.175    0.0250     0.10

Data are stored in coded form using these coding formulas ...
x1 ~ (Temp - 75)/5
x2 ~ (pH - 6)
```

Adding Response Variable

```
```{r}
library(rsm)
Y <- c(80.6,67.9,83.1,38.1,79.7,74.7,71.2,36.8,81.7,66.8,73.0,40.5,74.9,74.2,63.5,42.8,80.9,42.4,73.4,45.0,66.0,71.
7,77.5,76.3,67.4,86.5,77.4,74.6,79.8,78.3,74.8,80.9)
ccd1.df$Y <- Y
print(ccd1.df)
```
```

```
   run.order std.order Temp pH  redox naOxalate naLauryl    Y
1          1         1   70  5  0.075    0.0125     0.15 80.6
2          2         2   80  5  0.075    0.0125     0.05 67.9
3          3         3   70  7  0.075    0.0125     0.05 83.1
4          4         4   80  7  0.075    0.0125     0.15 38.1
5          5         5   70  5  0.275    0.0125     0.05 79.7
6          6         6   80  5  0.275    0.0125     0.15 74.7
7          7         7   70  7  0.275    0.0125     0.15 71.2
8          8         8   80  7  0.275    0.0125     0.05 36.8
9          9         9   70  5  0.075    0.0375     0.05 81.7
10        10        10   80  5  0.075    0.0375     0.15 66.8
11        11        11   70  7  0.075    0.0375     0.15 73.0
12        12        12   80  7  0.075    0.0375     0.05 40.5
13        13        13   70  5  0.275    0.0375     0.15 74.9
14        14        14   80  5  0.275    0.0375     0.05 74.2
15        15        15   70  7  0.275    0.0375     0.05 63.5
16        16        16   80  7  0.275    0.0375     0.15 42.8
17         1         1   65  6  0.175    0.0250     0.10 80.9
18         2         2   85  6  0.175    0.0250     0.10 42.4
19         3         3   75  4  0.175    0.0250     0.10 73.4
20         4         4   75  8  0.175    0.0250     0.10 45.0
21         5         5   75  6 -0.025    0.0250     0.10 66.0
22         6         6   75  6  0.375    0.0250     0.10 71.7
23         7         7   75  6  0.175    0.0000     0.10 77.5
24         8         8   75  6  0.175    0.0500     0.10 76.3
25         9         9   75  6  0.175    0.0250     0.00 67.4
26        10        10   75  6  0.175    0.0250     0.20 86.5
27        11        11   75  6  0.175    0.0250     0.10 77.4
28        12        12   75  6  0.175    0.0250     0.10 74.6
29        13        13   75  6  0.175    0.0250     0.10 79.8
30        14        14   75  6  0.175    0.0250     0.10 78.3
31        15        15   75  6  0.175    0.0250     0.10 74.8
32        16        16   75  6  0.175    0.0250     0.10 80.9

Data are stored in coded form using these coding formulas ...
x1 ~ (Temp - 75)/5
x2 ~ (pH - 6)
x3 ~ (redox - 0.175)/0.1
```

First Model - Coefficients

```
```{r}
library(rsm)
ccd1.model1 <- rsm(Y~SO(x1,x2,x3,x4,x5),data=ccd1.df)
summary(ccd1.model1)
```
```

```
Call:
rsm(formula = Y ~ SO(x1, x2, x3, x4, x5), data = ccd1.df)

             Estimate Std. Error  t value  Pr(>|t|)
(Intercept)  77.79432    1.85745  41.8823 1.750e-13 ***
x1          -10.12083    0.95058 -10.6470 3.938e-07 ***
x2           -8.67917    0.95058  -9.1304 1.821e-06 ***
x3           -0.10417    0.95058  -0.1096 0.9147145
x4           -0.71250    0.95058  -0.7495 0.4692583
x5            1.37083    0.95058   1.4421 0.1771350
x1:x2        -6.20625    1.16422  -5.3308 0.0002408 ***
x1:x3         2.76875    1.16422   2.3782 0.0366173 *
x1:x4         1.76875    1.16422   1.5193 0.1569046
x1:x5         0.70625    1.16422   0.6066 0.5564099
x2:x3        -1.68125    1.16422  -1.4441 0.1765837
x2:x4        -0.25625    1.16422  -0.2201 0.8298214
x2:x5         0.48125    1.16422   0.4134 0.6872818
x3:x4         0.04375    1.16422   0.0376 0.9706968
x3:x5         1.50625    1.16422   1.2938 0.2222458
x4:x5         0.03125    1.16422   0.0268 0.9790665
x1^2         -4.15682    0.85983  -4.8344 0.0005236 ***
x2^2         -4.76932    0.85983  -5.5468 0.0001736 ***
x3^2         -2.35682    0.85983  -2.7410 0.0191965 *
x4^2         -0.34432    0.85983  -0.4004 0.6964946
x5^2         -0.33182    0.85983  -0.3859 0.7069228
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

First Model - Anova

```
Analysis of Variance Table

Response: Y
                        Df Sum Sq Mean Sq F value    Pr(>F)
FO(x1, x2, x3, x4, x5)   5 4323.8  864.75 39.8751   1.1e-06
TWI(x1, x2, x3, x4, x5) 10  883.3   88.33  4.0730 0.0149793
PQ(x1, x2, x3, x4, x5)   5 1179.8  235.97 10.8809 0.0005774
Residuals               11  238.6   21.69
Lack of fit              6  205.5   34.24  5.1737 0.0458642
Pure error               5   33.1    6.62

Stationary point of response surface:
       x1        x2        x3        x4        x5
-0.394479 -0.624541  1.135045 -1.575681  3.694937

Stationary point in original units:
        Temp           pH        redox     naOxalate     naLauryl
73.027604848  5.375458965  0.288504539  0.005303989  0.284746826

Eigenanalysis:
eigen() decomposition
$values
[1]  0.5974405 -0.2085975 -1.0427378 -3.6500412 -7.6551549

$vectors
         [,1]        [,2]       [,3]       [,4]        [,5]
x1  0.4752854 -0.10058397  0.3809698  0.3867462  0.68503720
x2 -0.3351535  0.09775804 -0.4170033 -0.4292114  0.72111121
x3  0.4385796  0.15056861  0.3971482 -0.7899316 -0.05708322
x4  0.5097058 -0.63801217 -0.5611295 -0.1156766 -0.06995046
x5  0.4578470  0.74202143 -0.4564836  0.1697999 -0.05070604
```

Model Refinement

- Used $\alpha = 0.05$ for decision making
- **Terms to Keep** (8)
  - x1, x2, x3, x1:x2, x1:x3, x1^2, x2^2, x3^2
- **Terms to Remove (12)**
  - x4, x5, x1:x4, x1:x5, x2:x3, x2:x4, x2:x5, x3:x4, x3:x5, x4:x5, x4^2, x5^2

Reduced Model

```
```{r}
library(rsm)
ccd1.model2 <- rsm(Y~FO(x1,x2,x3)+TWI(x1,x2,x3)+PQ(x1,x2,x3),data=ccd1.df)
summary(ccd1.model2)
```
```

```
Call:
rsm(formula = Y ~ FO(x1, x2, x3) + TWI(x1, x2, x3) + PQ(x1, x2,
    x3), data = ccd1.df)

             Estimate Std. Error  t value  Pr(>|t|)
(Intercept)  77.17019    1.32414  58.2794 < 2.2e-16 ***
x1          -10.12083    0.87166 -11.6110 7.466e-11 ***
x2           -8.67917    0.87166  -9.9571 1.305e-09 ***
x3           -0.10417    0.87166  -0.1195  0.905961
x1:x2        -6.20625    1.06756  -5.8135 7.542e-06 ***
x1:x3         2.76875    1.06756   2.5935  0.016581 *
x2:x3        -1.68125    1.06756  -1.5749  0.129562
x1^2         -4.10481    0.78337  -5.2399 2.953e-05 ***
x2^2         -4.71731    0.78337  -6.0218 4.632e-06 ***
x3^2         -2.30481    0.78337  -2.9422  0.007538 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Multiple R-squared:  0.9395,    Adjusted R-squared:  0.9147
F-statistic: 37.93 on 9 and 22 DF,  p-value: 2.369e-11

Analysis of Variance Table

Response: Y
                Df Sum Sq Mean Sq F value    Pr(>F)
FO(x1, x2, x3)   3 4266.5 1422.16 77.9912 7.018e-12
TWI(x1, x2, x3)  3  784.2  261.39 14.3345 2.144e-05
PQ(x1, x2, x3)   3 1173.6  391.22 21.4543 9.926e-07
Residuals       22  401.2   18.23
Lack of fit      5   69.4   13.87  0.7107    0.6237
```

Partial F-test

```
```{r}
anova(ccd1.model2,ccd1.model1)
```
```

```
Analysis of Variance Table

Model 1: Y ~ FO(x1, x2, x3) + TWI(x1, x2, x3) + PQ(x1, x2, x3)
Model 2: Y ~ FO(x1, x2, x3, x4, x5) + TWI(x1, x2, x3, x4, x5) + PQ(x1,
    x2, x3, x4, x5)
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     22 401.17
2     11 238.55 11    162.62 0.6817 0.7322
```