Maximizing the Flight Time of a Paper HelicopterUsing Response Surface Methodology

Statistics

Dipl.-Ing. Dr. techn. Bernhard Spangl

Student's name:

Solveig-Stella Billes

Marjan Aziminezhad

Date: 02.2023

1. Introduction

Response Surface Methodology (RSM) is a set of techniques used to understand and optimize systems. It is commonly applied in the design, development, and formulation of new products, as well as the improvement of existing ones. (Myers et al., 2009).The goal of the study described in the introduction is to use RSM to maximize the flight time of a paper helicopter. The response variable being studied is the flight time, and the predictor variables include the rotor length, rotor width, body length, and fold length. The other variables, such as the foot length, fold width, fold direction, and paper type, are also specified.

The study is conducted using a specific software, R Version 4.1.1 and R Studio Version 1.4.1717. The flight measurements are taken by dropping the paper helicopters from a ceiling height of approximately 2.6m and measuring the flight time using a mobile phone stopwatch. The results of the study are presented in the Appendix, and certain parts of the output that guided the decision-making process are presented within the text.

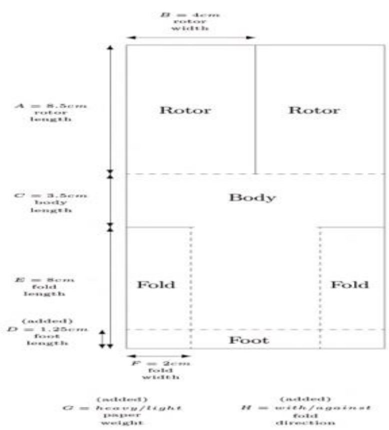

Figure 1:Blueprint of a helicopter, reproduced from [Erhardt, 2007]

2. Creating a design with 2^{k-1} elements

In this case, the study utilizes a complete factorial design with two levels (high and low) of four chosen factors to build 16 paper helicopters. However, to obtain information about the main effects and low-order interactions, a fraction of the complete factorial experiment can be run if certain high-order interactions are assumed to be insignificant. This type of design, known as a fractional factorial design 2^{k-p}, is commonly used in industry for screening experiments. The study used a 24−1 fractional factorial design to investigate the main effects and two-way interactions, with replicates in two blocks. (Myers et al., 2009). The order of the runs in the blocks was randomized. The high and low values for the four factors were already determined in the article, and the measures of the four factors, their coded values, and the measured response are shown in Table 1.

Table 1:Screening experiment with uncoded and coded factor levels

run. order	std. order	block	rotor length (A) [cm]	rotor width (B) [cm]	body length (C) [cm]	fold length (E) [cm]	A coded	B coded	C coded	E coded	Time [s]
1	7	1	5,5	5	5,5	5	-1	1	1	-1	2,04
2	3	1	5,5	5	1,5	9	-1	1	-1	1	1,93
3	5	1	5,5	3	5,5	9	-1	-1	1	1	1,73
4	2	1	11,5	3	1,5	9	1	-1	-1	1	2,62
5	8	1	11,5	5	5,5	9	1	1	1	1	2,33
6	6	1	11,5	3	5,5	5	1	-1	1	-1	2,63
7	4	1	11,5	5	1,5	5	1	1	-1	-1	3,28
8	1	1	5,5	3	1,5	5	-1	-1	-1	-1	2,15
1	1	2	5,5	3	1,5	5	-1	-1	-1	-1	2,26
2	3	2	5,5	5	1,5	9	-1	1	-1	1	1,97
3	4	2	11,5	5	1,5	5	1	1	-1	-1	3,33
4	7	2	5,5	5	5,5	5	-1	1	1	-1	1,96
5	2	2	11,5	3	1,5	9	1	-1	-1	1	2,66
6	6	2	11,5	3	5,5	5	1	-1	1	-1	2,86
7	5	2	5,5	3	5,5	9	-1	-1	1	1	1,84
8	8	2	11,5	5	5,5	9	1	1	1	1	2,56

3. First Linear Response Surface Model

A statistical model was developed that includes both main effects and interactions between variables. The initial model had an overall significance level of p = 0.000000347 and an F-statistic of 57.09 based on 7 and 8 degrees of freedom. The coefficients and their T-statistics are presented in Table 2. The analysis of variance for the model showed an error sum of squares of 0, which resulted in an "infinite" Lack of Fit F-statistic. Due to the limited sample size of 8 for a model with four predictor variables, further optimization was conducted. In a real-world setting, such a value may indicate that the model is "overfitted" and not reliable for prediction. Factors A and E were chosen for further optimization while factor C, although significant, was not considered in the subsequent analysis.

Table 2 :Model coefficients of the first model, Response: Time

	Estimate	Std.Error	t-value	$Pr(>\|t\|)$
(Intercept)	2.380	0.020	101.200	0.000
x1 - factor A	0.400	0.020	16.950	**0.000**
x2 - factor B	0.040	0.020	1.724	0.120
x3 - factor C	-0.140	0.020	-5.970	0.000
x4 - factor E	-0.180	0.020	-7.610	**0.000**
x1:x2	0.050	0.020	2.150	0.060
x1:x3	-0.050	0.020	-2.040	0.080
x1:x4	-0.060	0.020	-2.630	**0.030**

4. Identifying the requirements for further optimization efforts

The second response surface model used only factors A and E as predictors and flight times as the response. The model's parameters were found to be acceptable, with a high overall F statistic and low p-value and significant individual coefficients at a significance level of 0.01. The determination and adjusted determination coefficients were both above 0.80, indicating that the model explains a substantial portion of the data variation. The Lack of Fit F-statistic was not significant at a significance level of 0.05. A contour plot was created (Figure 2) to illustrate the response increasing, with a color gradient (green indicating lower response, and orange indicating higher response). The model did not account for any non-linear relations in the data, but it identified a region where the response could be improved by adjusting only two factors. The gradient technique (steepest ascent method) was used to provide 6 estimates of improved flight time, and 6 new helicopters were built to test this assumption.

As illustrated in Table 3, The prediction was not verified, most likely due to inaccuracies in the time measurement in both the initial and follow-up experiments. As a result, the conditions from step 3 were used for the next step. The variables that were not manipulated were reset to the original values specified in the study (rotor width = 4cm, body length = 3.5cm) for this and subsequent steps.

Table 3:The relationship between helicopter design and performance as determined by ridge analysis.

distance (as default step size)	factor A - coded	factor E - coded	rotor length [cm]	fold length [cm]	estimated flight time [s]	actual flight time [s]
0	0.00	0.00	8.50	7.00	2.38	2.28
1	0.91	-0.41	11.24	6.18	2.82	2.73
2	1.82	-0.82	13.97	5.36	3.26	3.31
3	**2.74**	**-1.23**	**16.71**	**4.54**	**3.70**	**3.96**
4	3.65	-1.64	19.45	3.72	4.14	3.78
5	4.56	-2.05	22.18	2.90	4.57	4.13

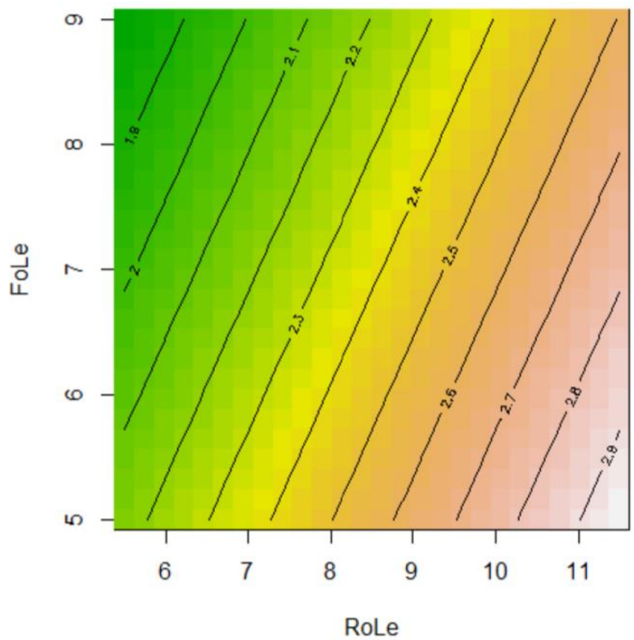

Figure 2 :Contour plot. This graph that illustrates the relationship between flight time and rotor length (x-axis) and fold length (y-axis) using contour lines

5. Central Composite Design and Second Order Model

Central composite designs are commonly used basic experimental designs for estimating second-order response surface models. These designs include a factorial design for estimating first-order and two-factor interactions, k axial (or "star") points for estimating the second-order terms, and replicated center points to estimate the second-order terms and estimate error. The value of α is set to $\sqrt{k}$ for rotatability, ensuring that the accuracy of prediction with a quadratic equation is not dependent on direction (Erhardt, 2007). A central composite design was generated with a center point of rotor length = 16.7cm and fold length = 4.5cm. The center point was repeated twice in the "cube" block and twice in the "star" block. The measurements of the resulting helicopters, their coded values, and the measured response are shown in Table 4. The second-order model created after the central composite design is acceptable, with a model F-statistic of 7.86 on 5 and 6 degrees of freedom (p-value of 0.013), and determination and adjusted determination coefficients both above 0.75. The individual model coefficients can be found in the appendix. Both factors were individually significant at $\alpha = 0.05$, as well as the quadratic term of the fold length.

Table 4 :Optimization experiment with coded and uncoded factor levels

run.order	std.order	block	rotor lenght A [cm]	fold lenght E [cm]	A coded	E coded	Time
1	6	1	16,7	4,5	0	0	4,04
2	2	1	19,44	3,68	1	-1	3,87
3	3	1	13,96	5,32	-1	1	3,25
4	1	1	13,96	3,68	-1	-1	3,32
5	5	1	16,7	4,5	0	0	3,96
6	4	1	19,44	5,32	1	1	3,45
1	2	2	20,57	4,5	1,41	0	4,27
2	4	2	16,7	5,66	0	1,41	2,66
3	6	2	16,7	4,5	0	0	4,1
4	5	2	16,7	4,5	0	0	3,82
5	3	2	16,7	3,34	0	-1,41	3,9
6	1	2	12,83	4,5	-1,41	0	3,14

6. The Final Paper Helicopter and conclusion

The second-order model has a stationary point, which is a maximum, at a rotor length of 20cm and fold length of 4cm in the original units. The contour plot of this model is shown in Figure 3. The estimated flight time of 4.23 seconds, as determined by the R function canonical path, was relatively close to the actual flight time of the helicopter, 4.17 seconds. However, it is important to note the limitations of our experiment and suggest ways for improvement. The sample size was small and could be increased in future experiments. Additionally, including more repetition blocks could improve the precision of the models. The quality of flight time measurement could be enhanced by recording the experiments and basing the measurements on the frame rate of a camera instead of human eye and reaction time. Also, releasing the helicopters from a greater height could be considered. In conclusion, some improvement in flight time was achieved, but the small scale of our experiment does not allow us to confirm that the true maximum has been reached.

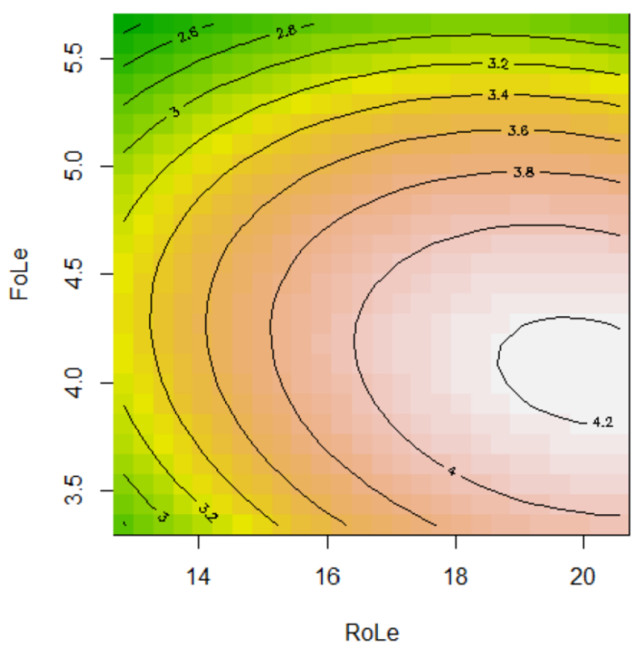

Figure 3 :The contour plot of flight time depending on rotor lenght (RoLe - x axis) and fold lenght

7. R Script

```
> library(rsm)
Warning message:
package ‘rsm’ was built under R version 4.1.2
>
> ## 1. Design
>
> ## basic design
> design <- cube(basis = ~ x1 + x2 + x3,
+                 generators = x4 ~ x1*x2*x3,
+                n0 = 0,
+                reps = 1,
+                 coding = c(x1 ~ (RoLe - 8.5)/3,
+                          x2 ~ (RoWi - 4.0)/1,
+                           x3 ~ (BoLe - 3.5)/2,
+                          x4 ~ (FoLe - 7.0)/2),
+                randomize = FALSE)
>
> as.data.frame(design)
  run.order std.order x1 x2 x3 x4
1         1         1 -1 -1 -1 -1
2         2         2  1 -1 -1  1
3         3         3 -1  1 -1  1
4         4         4  1  1 -1 -1
5         5         5 -1 -1  1  1
6         6         6  1 -1  1 -1
7         7         7 -1  1  1 -1
8         8         8  1  1  1  1
> design
  run.order std.order RoLe RoWi BoLe FoLe
1         1         1  5.5    3  1.5    5
2         2         2 11.5    3  1.5    9
3         3         3  5.5    5  1.5    9
4         4         4 11.5    5  1.5    5
5         5         5  5.5    3  5.5    9
6         6         6 11.5    3  5.5    5
7         7         7  5.5    5  5.5    5
8         8         8 11.5    5  5.5    9



x1 ~ (RoLe - 8.5)/3
x2 ~ (RoWi - 4)/1
x3 ~ (BoLe - 3.5)/2
x4 ~ (FoLe - 7)/2
>
> ## Setting up the experiment
>
> set.seed(8)     # setting your own seed
> expt <- djoin(dupe(design), dupe(design))
>
> as.data.frame(expt)
```

```
   run.order std.order x1 x2 x3 x4 Block
1          1         7 -1  1  1 -1     1
2          2         3 -1  1 -1  1     1
3          3         5 -1 -1  1  1     1
4          4         2  1 -1 -1  1     1
5          5         8  1  1  1  1     1
6          6         6  1 -1  1 -1     1
7          7         4  1  1 -1 -1     1
8          8         1 -1 -1 -1 -1     1
9          1         1 -1 -1 -1 -1     2
10         2         3 -1  1 -1  1     2
11         3         4  1  1 -1 -1     2
12         4         7 -1  1  1 -1     2
13         5         2  1 -1 -1  1     2
14         6         6  1 -1  1 -1     2
15         7         5 -1 -1  1  1     2
16         8         8  1  1  1  1     2
> expt
   run.order std.order RoLe RoWi BoLe FoLe Block
1          1         7  5.5    5  5.5    5     1
2          2         3  5.5    5  1.5    9     1
3          3         5  5.5    3  5.5    9     1
4          4         2 11.5    3  1.5    9     1
5          5         8 11.5    5  5.5    9     1
6          6         6 11.5    3  5.5    5     1
7          7         4 11.5    5  1.5    5     1
8          8         1  5.5    3  1.5    5     1
9          1         1  5.5    3  1.5    5     2
10         2         3  5.5    5  1.5    9     2
11         3         4 11.5    5  1.5    5     2
12         4         7  5.5    5  5.5    5     2
13         5         2 11.5    3  1.5    9     2
14         6         6 11.5    3  5.5    5     2
15         7         5  5.5    3  5.5    9     2
16         8         8 11.5    5  5.5    9     2

Data are stored in coded form using these coding formulas ...
x1 ~ (RoLe - 8.5)/3
x2 ~ (RoWi - 4)/1
x3 ~ (BoLe - 3.5)/2
x4 ~ (FoLe - 7)/2
>
> ## Writing CSV-File for measurements
>
> expt.coded <- code2val(expt, attr(expt, "codings"))
> write.csv2(cbind(expt.coded[, c(1:2, 7, 3:6)], expt[, 3:6]),
+            file="helicopter.csv", row.names=FALSE)
>
> ## 1. Design - re-generating the used design
>
> ## basic design
> design <- cube(basis = ~ x1 + x2 + x3,
+                generators = x4 ~ x1*x2*x3,
+               n0 = 0,
```

```
+                 reps = 1,
+                  coding = c(x1 ~ (RoLe - 8.5)/3,
+                             x2 ~ (RoWi - 4.0)/1,
+                              x3 ~ (BoLe - 3.5)/2,
+                             x4 ~ (FoLe - 7.0)/2),
+                 randomize = FALSE)
>
> ##  Setting up the experiment
>
> set.seed(008)      # setting your own seed
> expt <- djoin(dupe(design), dupe(design))
>
> expt
   run.order std.order RoLe RoWi BoLe FoLe Block
1           1          7   5.5    5   5.5    5     1
2           2          3   5.5    5   1.5    9     1
3           3          5   5.5    3   5.5    9     1
4           4          2  11.5    3   1.5    9     1
5           5          8  11.5    5   5.5    9     1
6           6          6  11.5    3   5.5    5     1
7           7          4  11.5    5   1.5    5     1
8           8          1   5.5    3   1.5    5     1
9           1          1   5.5    3   1.5    5     2
10          2          3   5.5    5   1.5    9     2
11          3          4  11.5    5   1.5    5     2
12          4          7   5.5    5   5.5    5     2
13          5          2  11.5    3   1.5    9     2
14          6          6  11.5    3   5.5    5     2
15          7          5   5.5    3   5.5    9     2

16          8          8  11.5    5   5.5    9     2

Data are stored in coded form using these coding formulas
... x1 ~ (RoLe - 8.5)/3
x2 ~ (RoWi - 4)/1
x3 ~ (BoLe - 3.5)/2
x4 ~ (FoLe - 7)/2
> dat <- read.csv2("helicopter_Marjan_Stella.csv")
>
> expt$Time <- dat$Time
>
> ##  First order and two-way interaction model of all data
> rsm1 <- rsm(Time ~ FO(x1, x2, x3, x4) + TWI(x1, x2, x3, x4),
+              data=expt)
> summary(rsm1)
Near-stationary-ridge situation detected -- stationary point altered
 Change 'threshold' if this is not what you intend

Call:
              Estimate Std. Error  t value  Pr(>|t|)
(Intercept)   2.384375   0.023560 101.2040 1.015e-13 ***
x1            0.399375   0.023560  16.9513 1.488e-07 ***
x2            0.040625   0.023560   1.7243 0.1229384
x3           -0.140625   0.023560  -5.9688 0.0003349 ***
x4           -0.179375   0.023560  -7.6135 6.226e-05 ***
x1:x2         0.050625   0.023560   2.1488 0.0638986 .
x1:x3        -0.048125   0.023560  -2.0426 0.0753634 .
x1:x4        -0.061875   0.023560  -2.6263 0.0303538 *
```

```
rsm(formula = Time ~ FO(x1, x2, x3, x4) + TWI(x1, x2, x3,
    x4), data = expt)


Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Multiple R-squared:  0.9804,Adjusted R-squared:  0.9632
F-statistic: 57.09 on 7 and 8 DF,  p-value: 3.347e-

06 Analysis of Variance Table

Response: Time
                    Df Sum Sq Mean Sq F value    Pr(>F)
FO(x1, x2, x3, x4)   4 3.4096       1 95.9782 8.539e-07
TWI(x1, x2, x3, x4)  3 0.1393       0  5.2289   0.02734
Residuals            8 0.0711       0
Lack of fit          0 0.0000     Inf
Pure error           8 0.0711       0

Stationary point of response surface:


       x1        x2        x3        x4
-2.288053 -2.321969  2.207303  2.837962

Stationary point in original units: RoLe
     RoWi      BoLe      FoLe
 1.635840  1.678031  7.914607 12.675923

Eigenanalysis:
eigen() decomposition
$values
[1]  0.04665678  0.00000000  0.00000000 -0.04665678

$vectors
         [,1]       [,2]       [,3]       [,4]
x1  0.7071068  0.0000000  0.0000000  0.7071068
x2  0.3836235  0.6630869  0.5157342 -0.3836235
x3 -0.3646792 -0.2216991  0.8275673  0.3646792
x4 -0.4688732  0.7149583 -0.2216991  0.4688732

>

> ##  3. Optimization (part 1)
>
> rsm2 <- rsm(Time ~ FO(x1, x4), data=expt)
> summary(rsm2)

Call:
             Estimate Std. Error t value  Pr(>|t|)
(Intercept)  2.384375   0.051571 46.2352 8.254e-16 ***
x1           0.399375   0.051571  7.7442 3.186e-06 ***
x4          -0.179375   0.051571 -3.4782  0.004081 **
---
rsm(formula = Time ~ FO(x1, x4), data = expt)
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Multiple R-squared:  0.8472,Adjusted R-squared:  0.8237
F-statistic: 36.04 on 2 and 13 DF,  p-value: 4.978e-

06 Analysis of Variance Table

Response: Time
             Df  Sum Sq Mean Sq F value    Pr(>F)
FO(x1, x4)    2 3.06681 1.53341 36.0357 4.978e-06
Residuals    13 0.55318 0.04255
Lack of fit   1 0.06126 0.06126  1.4943     0.245
Pure error   12 0.49192 0.04099


Direction of steepest ascent (at radius 1):
        x1         x4
 0.9122151 -0.4097116

Corresponding increment in original units:
      RoLe       FoLe
 2.7366453 -0.8194233

>
>
> ## Plotting
> contour(rsm2, ~ x1 + x4, image=TRUE)
>
> ## Optimization according to steepest ascent
> ## 5-step rediction accoring to first-order model
> # step size left as default, cause it produced non-negative lenghts
> steepest(rsm2, dist=0:5)
Path of steepest ascent from ridge analysis:
  dist    x1     x4 |   RoLe  FoLe |  yhat
1    0 0.000  0.000 |  8.500 7.000 | 2.384
2    1 0.912 -0.410 | 11.236 6.180 | 2.822
3    2 1.824 -0.819 | 13.972 5.362 | 3.260
4    3 2.737 -1.229 | 16.711 4.542 | 3.698
5    4 3.649 -1.639 | 19.447 3.722 | 4.136
6    5 4.561 -2.049 | 22.183 2.902 | 4.573
>
>
> #Corresponding increment in original units:
> #  RoLe       FoLe
> #2.7366453 -0.8194233
>
> ## 4. Optimization (part 2)
>
> ## Basic RSM design
> design2 <- ccd(2,
+               n0=2,
+                 coding = c(x1 ~ (RoLe - 16.7)/2.74,
+                            x2 ~ (FoLe - 4.5)/0.82),
+               randomize = FALSE)
>
> as.data.frame(design2)
   run.order std.order        x1        x2 Block
```

```
1        1        1  -1.000000  -1.000000     1
2        2        2   1.000000  -1.000000     1
3        3        3  -1.000000   1.000000     1
4        4        4   1.000000   1.000000     1
5        5        5   0.000000   0.000000     1
6        6        6   0.000000   0.000000     1
7        1        1  -1.414214   0.000000     2
8        2        2   1.414214   0.000000     2
9        3        3   0.000000  -1.414214     2
10       4        4   0.000000   1.414214     2
11       5        5   0.000000   0.000000     2
12       6        6   0.000000   0.000000     2
> design2
   run.order  std.order     RoLe      FoLe  Block
1        1        1  13.96000  3.680000     1
2        2        2  19.44000  3.680000     1
3        3        3  13.96000  5.320000     1
4        4        4  19.44000  5.320000     1
5        5        5  16.70000  4.500000     1
6        6        6  16.70000  4.500000     1
7        1        1  12.82505  4.500000     2
8        2        2  20.57495  4.500000     2
9        3        3  16.70000  3.340345     2
10       4        4  16.70000  5.659655     2
11       5        5  16.70000  4.500000     2
12       6        6  16.70000  4.500000     2

Data are stored in coded form using these coding formulas
... x1 ~ (RoLe - 16.7)/2.74
x2 ~ (FoLe - 4.5)/0.82
>
> expt2.coded <- code2val(expt2, attr(expt2, "codings"))
> write.csv2(cbind(expt2.coded[, c(1:2, 5)],
+                  round(expt2.coded[, 3:4], digits=2),
+                  round(expt2[, 3:4], digits=2)),
+            file="helicopter2_e_b.csv", row.names=FALSE)
>
> ## Basic RSM design
> design2 <- ccd(2,
+               n0=2,
+                coding = c(x1 ~ (RoLe - 16.7)/2.74,
+                           x2 ~ (FoLe - 4.5)/0.82),
+               randomize = FALSE)
>
> set.seed(8)    # setting your own seed
> expt2 <- dupe(design2)
>
> expt2
   run.order  std.order     RoLe      FoLe  Block
1        1        6  16.70000  4.500000     1


2        2        2  19.44000  3.680000     1
3        3        3  13.96000  5.320000     1
4        4        1  13.96000  3.680000     1
5        5        5  16.70000  4.500000     1
6        6        4  19.44000  5.320000     1
7        1        2  20.57495  4.500000     2
```

```
8              2              4  16.70000  5.659655        2
9              3              6  16.70000  4.500000        2
10             4              5  16.70000  4.500000        2
11             5              3  16.70000  3.340345        2
12             6              1  12.82505  4.500000        2

Data are stored in coded form using these coding formulas
... x1 ~ (RoLe - 16.7)/2.74
x2 ~ (FoLe - 4.5)/0.82
>
> ##  5. Analysis - Optimization (part 3)
>
> dat2 <- read.csv2("helicopter2_Marjan_stella_measurements.csv")
>
> expt2$Time <- dat2$Time
>
> rsm3 <- rsm(Time ~ SO(x1, x2), data=expt2)
> summary(rsm3)

Call:
              Estimate   Std. Error t value   Pr(>|t|)
(Intercept)   3.980000     0.118021  33.7228  4.526e-08 ***
x1            0.293508     0.083453   3.5170   0.012564 *
x2           -0.280453     0.083453  -3.3606   0.015218 *
x1:x2        -0.087500     0.118021  -0.7414   0.486448
x1^2         -0.142500     0.093304  -1.5273   0.177551
x2^2         -0.355000     0.093304  -3.8048   0.008918 **
---
rsm(formula = Time ~ SO(x1, x2), data = expt2)


Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Multiple R-squared:  0.8676,Adjusted R-squared:  0.7572
F-statistic:  7.86 on 5 and 6 DF,  p-value:

0.01304 Analysis of Variance Table

Response: Time
             Df  Sum Sq Mean Sq  F value    Pr(>F)
 FO(x1, x2)   2 1.31841 0.65920  11.8315  0.008276
 TWI(x1, x2)  1 0.03063 0.03063   0.5497  0.486448
 PQ(x1, x2)   2 0.84064 0.42032   7.5440  0.023033



Residuals     6 0.33429 0.05572
Lack of fit   3 0.29029 0.09676  6.5976  0.077785
Pure error    3 0.04400 0.01467

Stationary point of response surface:
        x1         x2
 1.1963922 -0.5424471

Stationary point in original units:
     RoLe      FoLe
19.978115  4.055193
```

```
Eigenanalysis: eigen()
decomposition
$values
[1] -0.1338451 -0.3636549

$vectors
            [,1]      [,2]


x1 -0.9809888 0.1940643
x2  0.1940643 0.9809888

>
> ##  Plotting
> contour(rsm3, ~ x1 + x2, image=TRUE)
>
> #choosing best measurements for helicopter
> canonical.path(rsm3, dist=seq(-5, 5, by=0.5), which=1)
   dist     x1     x2 |     RoLe     FoLe |  yhat
1  -5.0  6.101 -1.513 | 33.41674 3.25934 | 0.886
2  -4.5  5.611 -1.416 | 32.07414 3.33888 | 1.521
3  -4.0  5.120 -1.319 | 30.72880 3.41842 | 2.090
4  -3.5  4.630 -1.222 | 29.38620 3.49796 | 2.592
5  -3.0  4.139 -1.125 | 28.04086 3.57750 | 3.027
6  -2.5  3.649 -1.028 | 26.69826 3.65704 | 3.395
7  -2.0  3.158 -0.931 | 25.35292 3.73658 | 3.696
8  -1.5  2.668 -0.834 | 24.01032 3.81612 | 3.930
9  -1.0  2.177 -0.737 | 22.66498 3.89566 | 4.098
10 -0.5  1.687 -0.639 | 21.32238 3.97602 | 4.198
11  0.0  1.196 -0.542 | 19.97704 4.05556 | 4.232
12  0.5  0.706 -0.445 | 18.63444 4.13510 | 4.198
13  1.0  0.215 -0.348 | 17.28910 4.21464 | 4.098
14  1.5 -0.275 -0.251 | 15.94650 4.29418 | 3.930
15  2.0 -0.766 -0.154 | 14.60116 4.37372 | 3.696
16  2.5 -1.256 -0.057 | 13.25856 4.45326 | 3.395
17  3.0 -1.747  0.040 | 11.91322 4.53280 | 3.027
18  3.5 -2.237  0.137 | 10.57062 4.61234 | 2.592
19  4.0 -2.728  0.234 |  9.22528 4.69188 | 2.090
20  4.5 -3.218  0.331 |  7.88268 4.77142 | 1.521
21  5.0 -3.709  0.428 |  6.53734 4.85096 | 0.885
> canonical.path(rsm3, dist=seq(-5, 5, by=0.5), which=2)
   dist    x1     x2 |     RoLe    FoLe |   yhat
1  -5.0 0.226 -5.447 | 17.31924 0.03346 | -4.858
2  -4.5 0.323 -4.957 | 17.58502 0.43526 | -3.133
3  -4.0 0.420 -4.466 | 17.85080 0.83788 | -1.586
4  -3.5 0.517 -3.976 | 18.11658 1.23968 | -0.223
5  -3.0 0.614 -3.485 | 18.38236 1.64230 |  0.960
6  -2.5 0.711 -2.995 | 18.64814 2.04410 |  1.959
7  -2.0 0.808 -2.504 | 18.91392 2.44672 |  2.778
8  -1.5 0.905 -2.014 | 19.17970 2.84852 |  3.413
9  -1.0 1.002 -1.523 | 19.44548 3.25114 |  3.868
10 -0.5 1.099 -1.033 | 19.71126 3.65294 |  4.141
11  0.0 1.196 -0.542 | 19.97704 4.05556 |  4.232
12  0.5 1.293 -0.052 | 20.24282 4.45736 |  4.141
13  1.0 1.390  0.439 | 20.50860 4.85998 |  3.868
14  1.5 1.487  0.929 | 20.77438 5.26178 |  3.414
15  2.0 1.585  1.420 | 21.04290 5.66440 |  2.776
16  2.5 1.682  1.910 | 21.30868 6.06620 |  1.959
17  3.0 1.779  2.401 | 21.57446 6.46882 |  0.958
```

18	3.5	1.876	2.891	|	21.84024	6.87062	|	-0.223
19	4.0	1.973	3.382	|	22.10602	7.27324	|	-1.588
20	4.5	2.070	3.872	|	22.37180	7.67504	|	-3.133
21	5.0	2.167	4.362	|	22.63758	8.07684	|	-4.858

8. References

Babyak MA. What you see may not be what you get: a brief, nontechnical introduction to overfitting in regression-type models. Psychosom Med. 2004 May-Jun; 66(3):411-21.

Erhard E. B. Designing a better paper helicopter using response surface methodology. Stats. 2007; 48: 14-21.

Lenth, R. V. Response-surface methods in R, using RSM. Journal of Statistical Software. 2009; 32(7), pp. 1–17.

Myers, Raymond H, Douglas C. Montgomery, and Christine M. Anderson-Cook. Response Surface Methodology: Process and Product Optimization Using Designed Experiments. Hoboken, N.J:Wiley, 2009.