Chapter 2

Building Empirical Models

2.1 Table 2.8, p. 48

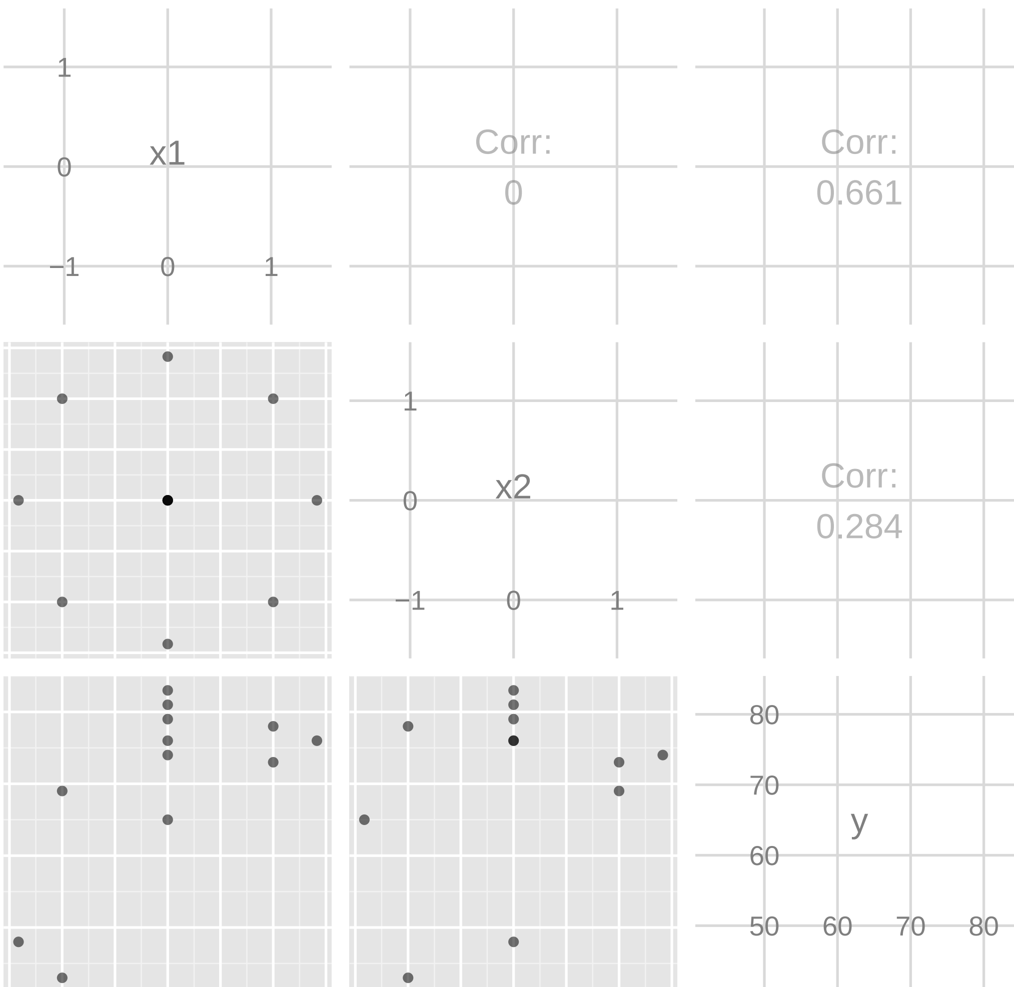
2.1.1 Read data

Read data, skip extra header lines, recode variables.

```
#### 2.8
fn.data <- "http://statacumen.com/teach/RSM/data/RSM_TAB_02-08.txt"
df.2.8 <- read.table(fn.data, header=TRUE, skip=1)
str(df.2.8)

## 'data.frame': 12 obs. of  3 variables:
##  $ x1: int  -1 1 -1 1 -9 9 0 0 0 0 ...
##  $ x2: int  -1 -1 1 1 0 0 -9 9 0 0 ...
##  $ y : int  43 78 69 73 48 76 65 74 76 79 ...

df.2.8

##    x1 x2  y
## 1  -1 -1 43
## 2   1 -1 78
## 3  -1  1 69
## 4   1  1 73
## 5  -9  0 48
## 6   9  0 76
## 7   0 -9 65
## 8   0  9 74
## 9   0  0 76
## 10  0  0 79
## 11  0  0 83
## 12  0  0 81

# replace coded values "9" with sqrt(2)
 # if x1= 9 then x1= sqrt(2)
df.2.8[,c("x1","x2")] <- replace(df.2.8[,c("x1","x2")], (df.2.8[,c("x1","x2")] ==  9)
                                 ,  sqrt(2))
df.2.8[,c("x1","x2")] <- replace(df.2.8[,c("x1","x2")], (df.2.8[,c("x1","x2")] == -9)
                                 , -sqrt(2))
df.2.8

##        x1     x2  y
## 1  -1.000 -1.000 43
## 2   1.000 -1.000 78
## 3  -1.000  1.000 69
## 4   1.000  1.000 73
## 5  -1.414  0.000 48
## 6   1.414  0.000 76
## 7   0.000 -1.414 65
## 8   0.000  1.414 74
## 9   0.000  0.000 76
## 10  0.000  0.000 79
## 11  0.000  0.000 83
## 12  0.000  0.000 81
```

Scatterplot matrix shows some relationships between y and other variables.

```
library(ggplot2)
suppressMessages(suppressWarnings(library(GGally)))
p <- ggpairs(df.2.8, alpha = 0.1)
# put scatterplots on top so y axis is vertical
#p <- ggpairs(df.2.8, upper = list(continuous = "points")
#                , lower = list(continuous = "cor")
#             )
print(p)

  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  detach("package:GGally", unload=TRUE)
  detach("package:reshape", unload=TRUE)

## Error:  invalid 'name' argument
```



Correlation matrix indicates some (linear) correlations with y are different than zero, but if curvature exists, this summary is not very meaningful.

```
# correlation matrix and associated p-values testing "H0: rho == 0"
library(Hmisc)
rcorr(as.matrix(df.2.8))

##      x1   x2    y
## x1 1.00 0.00 0.66
## x2 0.00 1.00 0.28
```

```
## y  0.66 0.28 1.00
##
## n= 12
##
##
## P
##    x1     x2     y
## x1        1.0000 0.0193
## x2 1.0000        0.3718
## y  0.0193 0.3718
```

2.1.2 Fit second-order model

Because this is a special kind of model (a full second-order model), we can get the test for higher order terms and lack of fit simply by using `rsm()`.

Fit second-order linear model.

```
# load the rsm package
library(rsm)

# fit second-order (SO) model
#   -- look up ?SO and see other options
rsm.2.8.y.SOx12 <- rsm(y ~ SO(x1, x2), data = df.2.8)

# which variables are available in the rsm object?
names(rsm.2.8.y.SOx12)

##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
## [13] "data"          "b"             "order"         "B"
## [17] "newlabs"

# which variables are available in the summary of the rsm object?
names(summary(rsm.2.8.y.SOx12))

##  [1] "call"          "terms"         "residuals"     "coefficients"
##  [5] "aliased"       "sigma"         "df"            "r.squared"
##  [9] "adj.r.squared" "fstatistic"    "cov.unscaled"  "canonical"
## [13] "lof"

# show the summary
summary(rsm.2.8.y.SOx12)

##
## Call:
## rsm(formula = y ~ SO(x1, x2), data = df.2.8)
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   79.750      1.214   65.71  8.4e-10 ***
## x1             9.825      0.858   11.45  2.7e-05 ***
## x2             4.216      0.858    4.91  0.00268 **
## x1:x2         -7.750      1.214   -6.39  0.00069 ***
```

```
## x1^2          -8.875      0.959   -9.25  9.0e-05 ***
## x2^2          -5.125      0.959   -5.34  0.00176 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared:  0.98,Adjusted R-squared:  0.963
## F-statistic: 58.9 on 5 and 6 DF,  p-value: 5.12e-05
##
## Analysis of Variance Table
##
## Response: y
##              Df Sum Sq Mean Sq F value  Pr(>F)
## FO(x1, x2)   2    914     457   77.61 5.2e-05
## TWI(x1, x2)  1    240     240   40.78 0.00069
## PQ(x1, x2)   2    579     289   49.13 0.00019
## Residuals    6     35       6
## Lack of fit  3      9       3    0.32 0.81192
## Pure error   3     27       9
##
## Stationary point of response surface:
##       x1       x2
##  0.55819 -0.01073
##
## Eigenanalysis:
## $values
## [1]  -2.695 -11.305
##
## $vectors
##       [,1]    [,2]
## x1  0.5312 -0.8472
## x2 -0.8472 -0.5312
# include externally Studentized residuals in the rsm object for plotting later
rsm.2.8.y.SOx12$studres <- rstudent(rsm.2.8.y.SOx12)
```

2.1.3 Model fitting

The following illustrates fitting several model types using `rsm()`.

Fit a first-order model.

```
# fit the first-order model
rsm.2.8.y.FOx12 <- rsm(y ~ FO(x1, x2), data = df.2.8)
# externally Studentized residuals
rsm.2.8.y.FOx12$studres <- rstudent(rsm.2.8.y.FOx12)
summary(rsm.2.8.y.FOx12)
##
## Call:
## rsm(formula = y ~ FO(x1, x2), data = df.2.8)
##
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)    70.42       2.81   25.03  1.2e-09 ***
## x1              9.82       3.45    2.85    0.019 *
## x2              4.22       3.45    1.22    0.252
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared:  0.517,Adjusted R-squared:  0.41
## F-statistic: 4.82 on 2 and 9 DF,  p-value: 0.0378
##
## Analysis of Variance Table
##
## Response: y
##             Df Sum Sq Mean Sq F value Pr(>F)
## FO(x1, x2)   2    914     457    4.82  0.038
## Residuals    9    855      95
## Lack of fit  6    828     138   15.47  0.023
## Pure error   3     27       9
##
## Direction of steepest ascent (at radius 1):
##     x1     x2
## 0.9190 0.3943
##
## Corresponding increment in original units:
##     x1     x2
## 0.9190 0.3943
```

Fit a first-order with two-way interaction model

```
# fit the first-order with two-way interaction model.
rsm.2.8.y.TWIx12 <- rsm(y ~ FO(x1, x2) + TWI(x1, x2), data = df.2.8)
# externally Studentized residuals
rsm.2.8.y.TWIx12$studres <- rstudent(rsm.2.8.y.TWIx12)
summary(rsm.2.8.y.TWIx12)
```

```
##
## Call:
## rsm(formula = y ~ FO(x1, x2) + TWI(x1, x2), data = df.2.8)
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    70.42       2.53   27.84    3e-09 ***
## x1              9.82       3.10    3.17    0.013 *
## x2              4.22       3.10    1.36    0.211
## x1:x2          -7.75       4.38   -1.77    0.115
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared:  0.653,Adjusted R-squared:  0.523
## F-statistic: 5.01 on 3 and 8 DF,  p-value: 0.0304
##
## Analysis of Variance Table
##
## Response: y
##             Df Sum Sq Mean Sq F value Pr(>F)
## FO(x1, x2)   2    914     457    5.95  0.026
```

```
## TWI(x1, x2)  1    240     240    3.13  0.115
## Residuals    8    614      77
## Lack of fit  5    588     118   13.18  0.030
## Pure error   3     27       9
##
## Stationary point of response surface:
##    x1    x2
## 0.544 1.268
##
## Eigenanalysis:
## $values
## [1]  3.875 -3.875
##
## $vectors
##       [,1]    [,2]
## x1 -0.7071 -0.7071
## x2  0.7071 -0.7071
```

Fit a second-order without interactions model.

```
# Fit the second-order without interactions model
rsm.2.8.y.PQx12 <- rsm(y ~ FO(x1, x2) + PQ(x1, x2), data = df.2.8)
# externally Studentized residuals
rsm.2.8.y.PQx12$studres <- rstudent(rsm.2.8.y.PQx12)
summary(rsm.2.8.y.PQx12)
```

```
##
## Call:
## rsm(formula = y ~ FO(x1, x2) + PQ(x1, x2), data = df.2.8)
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    79.75       3.14   25.42  3.7e-08 ***
## x1              9.82       2.22    4.43    0.003 **
## x2              4.22       2.22    1.90    0.099 .
## x1^2           -8.87       2.48   -3.58    0.009 **
## x2^2           -5.12       2.48   -2.07    0.078 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared:  0.844,Adjusted R-squared:  0.755
## F-statistic: 9.48 on 4 and 7 DF,  p-value: 0.0059
##
## Analysis of Variance Table
##
## Response: y
##             Df Sum Sq Mean Sq F value Pr(>F)
## FO(x1, x2)   2    914     457   11.61  0.006
## PQ(x1, x2)   2    579     289    7.35  0.019
## Residuals    7    276      39
## Lack of fit  4    249      62    6.98  0.071
## Pure error   3     27       9
##
## Stationary point of response surface:
##     x1     x2
```

```
## 0.5535 0.4113
##
## Eigenanalysis:
## $values
## [1] -5.125 -8.875
##
## $vectors
##    [,1] [,2]
## x1    0   -1
## x2   -1    0
```

```
## There are at least two ways of specifying the full second-order model
#  SO(x1, x2) = FO(x1, x2) + TWI(x1, x2) + PQ(x1, x2)
#              = x1 + x2 + x1:x2 + x1^2 + x2^2
#              = (x1 + x2)^2
```

2.1.4 Model comparisons

Model comparison (for multi-parameter tests).

```
# compare the reduced first-order model to the full second-order model
anova(rsm.2.8.y.FOx12, rsm.2.8.y.SOx12)
```

```
## Analysis of Variance Table
##
## Model 1: y ~ FO(x1, x2)
## Model 2: y ~ FO(x1, x2) + TWI(x1, x2) + PQ(x1, x2)
##   Res.Df RSS Df Sum of Sq    F  Pr(>F)
## 1      9 855
## 2      6  35  3       819 46.4 0.00015 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Confidence intervals and prediction intervals.

```
# conf int for parameters
confint(rsm.2.8.y.SOx12)
```

```
##                   2.5 % 97.5 %
## (Intercept)      76.780 82.720
## FO(x1, x2)x1      7.725 11.925
## FO(x1, x2)x2      2.116  6.316
## TWI(x1, x2)     -10.720 -4.780
## PQ(x1, x2)x1^2 -11.223 -6.527
## PQ(x1, x2)x2^2  -7.473 -2.777
```

```
# conf int for regression line
predict(rsm.2.8.y.SOx12, df.2.8[1:dim(df.2.8)[1],], interval = "confidence")
```

```
##     fit   lwr   upr
## 1  43.96 39.26 48.65
## 2  79.11 74.41 83.80
## 3  67.89 63.20 72.59
## 4  72.04 67.35 76.74
## 5  48.11 43.41 52.80
```

```
## 6  75.89 71.20 80.59
## 7  63.54 58.84 68.23
## 8  75.46 70.77 80.16
## 9  79.75 76.78 82.72
## 10 79.75 76.78 82.72
## 11 79.75 76.78 82.72
## 12 79.75 76.78 82.72
```

```
# pred int for new observations
predict(rsm.2.8.y.SOx12, df.2.8[1:dim(df.2.8)[1],], interval = "prediction")
```

```
##      fit   lwr   upr
## 1  43.96 36.39 51.53
## 2  79.11 71.54 86.68
## 3  67.89 60.32 75.46
## 4  72.04 64.47 79.61
## 5  48.11 40.53 55.68
## 6  75.89 68.32 83.47
## 7  63.54 55.97 71.11
## 8  75.46 67.89 83.03
## 9  79.75 73.11 86.39
## 10 79.75 73.11 86.39
## 11 79.75 73.11 86.39
## 12 79.75 73.11 86.39
```

2.1.5 Lack-of-fit test

The lack-of-fit (LOF) test is equivalent to a comparison between two models. First, we define the full model by setting up a categorical group variable that will take unique values for each distinct pair of (x1, x2) values. This group variable fits a model that is equivalent to a one-way ANOVA. The SSE for the full model is 26.75 with 3 df. This is the pure error SS and df. (Try this for yourself.) Second, we define the reduced model (reduced compared to the one-way ANOVA above) as the regression model fit (taking x1 and x2 as continuous variables). The SSE for the reduced model is 35.35 with 6 df. This is the residual SS. The LOF SS is the difference of SSE between the reduced and the full model: $35.35 - 26.75 = 8.6$ with $6 - 3 = 3$ df. The F-test is then the LOF SSE and df vs the full SSE and df, $F = (8.6/3)/(26.75/3) = 0.32$, where there are 3 df and 3 df in the numerator and denominator

```
summary(rsm.2.8.y.SOx12)$lof
```

```
## Analysis of Variance Table
##
## Response: y
##             Df Sum Sq Mean Sq F value  Pr(>F)
```

```
## FO(x1, x2)   2    914     457   77.61 5.2e-05 ***
## TWI(x1, x2)  1    240     240   40.78 0.00069 ***
## PQ(x1, x2)   2    579     289   49.13 0.00019 ***
## Residuals    6     35       6
## Lack of fit  3      9       3    0.32 0.81192
## Pure error   3     27       9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2.1.6 Diagnostics and plots of estimated response surfaces

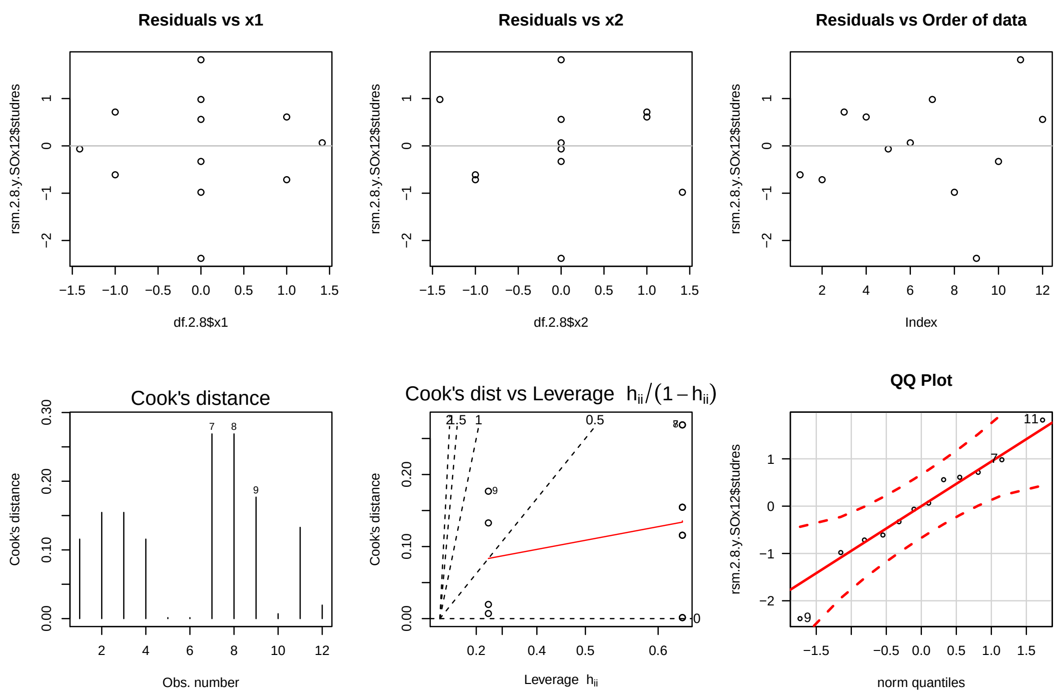
Plot the residuals, and comment on model adequacy.

```
# plot diagnostics
par(mfrow=c(2,4))

plot(df.2.8$x1, rsm.2.8.y.SOx12$studres, main="Residuals vs x1")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(df.2.8$x2, rsm.2.8.y.SOx12$studres, main="Residuals vs x2")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# residuals vs order of data
plot(rsm.2.8.y.SOx12$studres, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(rsm.2.8.y.SOx12, which = c(1,4,6))

# Normality of Residuals
library(car)
qqPlot(rsm.2.8.y.SOx12$studres, las = 1, id.n = 3, main="QQ Plot")
```

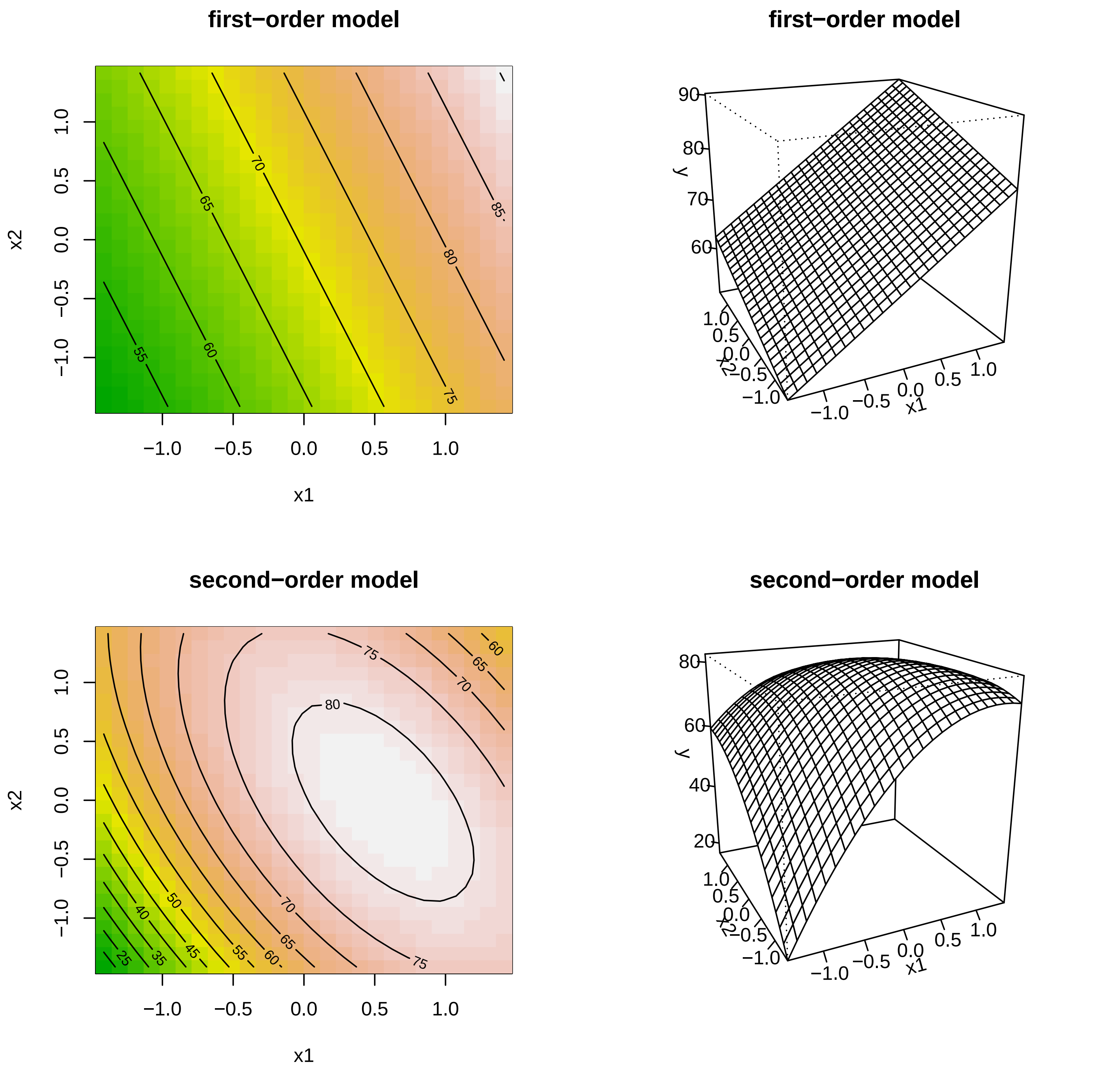
```
##  9 11  7
##  1 12 11
```

```
cooks.distance(rsm.2.8.y.SOx12)
```

```
##        1        2        3        4        5        6        7
## 0.115698 0.154570 0.154570 0.115698 0.001405 0.001405 0.268863
##        8        9       10       11       12
## 0.268863 0.176813 0.007073 0.132806 0.019646
```




```
# first-order model
par(mfrow=c(2,2))
contour(rsm.2.8.y.FOx12, ~ x1 + x2, image = TRUE, main="first-order model")
persp(rsm.2.8.y.FOx12, x2 ~ x1, zlab = "y", main="first-order model")

# second-order model
contour(rsm.2.8.y.SOx12, ~ x1 + x2, image = TRUE, main="second-order model")
persp(rsm.2.8.y.SOx12, x2 ~ x1, zlab = "y", main="second-order model")
```


first−order model
x1
x2
first−order model
x1
x2
y
second−order model
x1
x2
second−order model
x1
x2
y