Metodologia de superfície de resposta: Uma introdução no software R

L9(Final) - Planejamento de experimentos II

19 de novembro de 2015

- Giovani Carrara Rodrigues 7151669

Resumo

A metodologia de superfície de resposta consiste em uma coleção de técnicas estatísticas e matemáticas útil para desenvolvimento, melhora e otimização de processos. Ela também tem aplicações importantes em planejamentos, desenvolvimento e formulação de novos produtos, e melhoria dos projetos e produtos existentes. A mais extensiva aplicação do RSM é na área industrial, particularmente em situações em que entram várias variáveis que potencialmente influenciam em alguma medida de desempenho ou na qualidade característica de um produto ou processo. E, essa medida de desempenho ou qualidade característica é chamada de resposta. (Myers e Montgomery, 1995).

Neste trabalho, mostraremos como aplicar a **RSM** nos software R em forma de exemplos.

1. Introdução

A metodologia de superficie de resposta é um método estatístico para a modelagem e análise de problemas nos quais a variável resposta é influenciada por vários fatores, onde buscamos a otimização dessa resposta. No software R usaremos o pacote rsm para aplicação do método. Vale resaltar, que a codificação apropriada dos dados é um fator importante para a análise de superfície de resposta, então o primeiro passo consiste em apresentar as codificações dos níveis dos fatores. Após esta etapa vamos ajustar os modelos e verificar a adequabilidade dos mesmos.

2. Metodologia

2.1 Codificação dos dados

Os dados que vamos utilizar possuem o nome de *ChemReact* (Tabela 7.6 de Myers; Montgomery; Anderson-Cook, 2009), e se encontra no pacote rsm.

```
# carregando o pacote
require(rsm)

# Dados na variável ChemReact
ChemReact
```

```
##     Time   Temp Block Yield
## 1  80.00 170.00    B1  80.5
## 2  80.00 180.00    B1  81.5
## 3  90.00 170.00    B1  82.0
```

```
## 4  90.00 180.00    B1  83.5
## 5  85.00 175.00    B1  83.9
## 6  85.00 175.00    B1  84.3
## 7  85.00 175.00    B1  84.0
## 8  85.00 175.00    B2  79.7
## 9  85.00 175.00    B2  79.8
## 10 85.00 175.00    B2  79.5
## 11 92.07 175.00    B2  78.4
## 12 77.93 175.00    B2  75.6
## 13 85.00 182.07    B2  78.5
## 14 85.00 167.93    B2  77.0
```

Primeiro vamos análisar somente o bloco 1 (B1). Observa-se que os valores dos fatores tempo(Time) e temperatura(Temp) do bloco 1 variam da seguinte forma 85 ± 5 e 175 ± 5 respectivamente, com 3 pontos centrais. Com isso temos as variáveis codificadas da seguinte maneira:

$X_1 = (Time - 85)/5 e X_2 = (Temp - 175)/5$

O pacote rsm possui várias funções que codificam as variáveis, nesse caso vamos usar a função *coded.data* para codificar os dados. Os valores só do bloco 1 são dados pelo conjunto de dados *ChemReact1* e os dados só com o bloco 2 pelo *ChemReact2*.

```
CR1 = coded.data(ChemReact1, x1 ~(Time - 85)/5, x2 ~(Temp - 175)/5)
CR1
```

```
##   Time Temp Yield
## 1   80  170  80.5
## 2   80  180  81.5
## 3   90  170  82.0
## 4   90  180  83.5
## 5   85  175  83.9
## 6   85  175  84.3
## 7   85  175  84.0
##
## Data are stored in coded form using these coding formulas ...
## x1 ~ (Time - 85)/5
## x2 ~ (Temp - 175)/5
```

Esse listagem parace muito com os dados originais, mas internamente, os dados são salvos na forma codificada como podemos ver através do seguinte "truque" de tranformar a variável em um *data.frame*.

```
as.data.frame(CR1)
```

```
##   x1 x2 Yield
## 1 -1 -1  80.5
## 2 -1  1  81.5
## 3  1 -1  82.0
## 4  1  1  83.5
## 5  0  0  83.9
## 6  0  0  84.3
## 7  0  0  84.0
```

2.2 Ajustando um modelo de superfície de resposta

Com os dados já codificados, no qual a produção (Yield) é a variável resposta. Vamos ajustar um modelo de primeira orderm.

```
CR <- coded.data(ChemReact, x1 ~(Time - 85)/5, x2 ~(Temp - 175)/5)

CR.rsm1 <- rsm(Yield~FO(x1,x2),data = CR,subset = (Block =="B1"))

summary(CR.rsm1)
```

```
##
## Call:
## rsm(formula = Yield ~ FO(x1, x2), data = CR, subset = (Block ==
##     "B1"))
##
##             Estimate Std. Error  t value  Pr(>|t|)
## (Intercept) 82.81429    0.54719 151.3456 1.143e-08 ***
## x1           0.87500    0.72386   1.2088    0.2933
## x2           0.62500    0.72386   0.8634    0.4366
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared:  0.3555, Adjusted R-squared:  0.0333
## F-statistic: 1.103 on 2 and 4 DF,  p-value: 0.4153
##
## Analysis of Variance Table
##
## Response: Yield
##             Df Sum Sq Mean Sq F value  Pr(>F)
## FO(x1, x2)   2 4.6250  2.3125  1.1033 0.41534
## Residuals    4 8.3836  2.0959
## Lack of fit  2 8.2969  4.1485 95.7335 0.01034
## Pure error   2 0.0867  0.0433
##
## Direction of steepest ascent (at radius 1):
##        x1        x2
## 0.8137335 0.5812382
##
## Corresponding increment in original units:
##     Time     Temp
## 4.068667 2.906191
```

Não há termos quadráticos e de interação no ajuste do modelo, vemos que as variáveis x1 e x2 não sáo significativas.

Há uma falta de ajuste significativo (p-valor = 0,01), vamos então testar um modelo de ordem maior. Incluindo também as interações.

```
 CR.rsm1.5 <- update(CR.rsm1,.~.+TWI(x1,x2))
summary(CR.rsm1.5)
```

```
##
```

```
## Call:
## rsm(formula = Yield ~ FO(x1, x2) + TWI(x1, x2), data = CR, subset = (Block ==
##     "B1"))
##
##             Estimate Std. Error  t value  Pr(>|t|)
## (Intercept) 82.81429    0.62948 131.5604 9.683e-07 ***
## x1           0.87500    0.83272   1.0508    0.3705
## x2           0.62500    0.83272   0.7506    0.5074
## x1:x2        0.12500    0.83272   0.1501    0.8902
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared:  0.3603, Adjusted R-squared:  -0.2793
## F-statistic: 0.5633 on 3 and 3 DF,  p-value: 0.6755
##
## Analysis of Variance Table
##
## Response: Yield
##             Df Sum Sq Mean Sq  F value   Pr(>F)
## FO(x1, x2)   2 4.6250  2.3125   0.8337 0.515302
## TWI(x1, x2)  1 0.0625  0.0625   0.0225 0.890202
## Residuals    3 8.3211  2.7737
## Lack of fit  1 8.2344  8.2344 190.0247 0.005221
## Pure error   2 0.0867  0.0433
##
## Stationary point of response surface:
## x1 x2
## -5 -7
##
## Stationary point in original units:
## Time Temp
##   60  140
##
## Eigenanalysis:
## $values
## [1]  0.0625 -0.0625
##
## $vectors
##         [,1]       [,2]
## x1 0.7071068 -0.7071068
## x2 0.7071068  0.7071068
```

Novamente vemos que a falta de ajuste é significativa, com p-valor = 0,01. Acrescentamos então os dados do bloco 2 a fim de montar um model de segunda ordem. Fazemos isso utilizando *SO(x1,x2)*, que inclui termos quadráticos e interação:

```
CR.rsm2 <- rsm(Yield ~ Block + SO(x1, x2), data = CR)
 summary(CR.rsm2)
```

```
##
## Call:
## rsm(formula = Yield ~ Block + SO(x1, x2), data = CR)
##
```

```
##              Estimate Std. Error  t value  Pr(>|t|)
## (Intercept) 84.095427   0.079631 1056.067 < 2.2e-16 ***
## BlockB2     -4.457530   0.087226  -51.103 2.877e-10 ***
## x1           0.932541   0.057699   16.162 8.444e-07 ***
## x2           0.577712   0.057699   10.012 2.122e-05 ***
## x1:x2        0.125000   0.081592    1.532    0.1694
## x1^2        -1.308555   0.060064  -21.786 1.083e-07 ***
## x2^2        -0.933442   0.060064  -15.541 1.104e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared:  0.9981, Adjusted R-squared:  0.9964
## F-statistic: 607.2 on 6 and 7 DF,  p-value: 3.811e-09
##
## Analysis of Variance Table
##
## Response: Yield
##             Df Sum Sq Mean Sq   F value    Pr(>F)
## Block        1 69.531  69.531 2611.0950 2.879e-10
## FO(x1, x2)   2  9.626   4.813  180.7341 9.450e-07
## TWI(x1, x2)  1  0.063   0.063    2.3470    0.1694
## PQ(x1, x2)   2 17.791   8.896  334.0539 1.135e-07
## Residuals    7  0.186   0.027
## Lack of fit  3  0.053   0.018    0.5307    0.6851
## Pure error   4  0.133   0.033
##
## Stationary point of response surface:
##        x1        x2
## 0.3722954 0.3343802
##
## Stationary point in original units:
##      Time      Temp
##  86.86148 176.67190
##
## Eigenanalysis:
## $values
## [1] -0.9233027 -1.3186949
##
## $vectors
##          [,1]       [,2]
## x1 -0.1601375 -0.9870947
## x2 -0.9870947  0.1601375
```

Observamos que agora a falta de ajuste é não significativo (p-valor = 0,69). O ponto de estacionariedade do modelo, isto é, ponto de máximo esta em (0,37;0,33), já que os autovalores deram negativos. temos que o ideal estimado é Tempo = 17 e Temperatura = 177.

3. Suposições do modelo

Verificar as suposições do modelo é esssencial para a análise ter validade.

3.1 Normalidade

Quando ajustamos o modelo acima o valor que a variável *CR.rsm2* quarda é um objeto com o nome de *lista* no R. Essa *lista* contém vários tipos de variáveis, e para observá-las, usamos o seguinte comando.

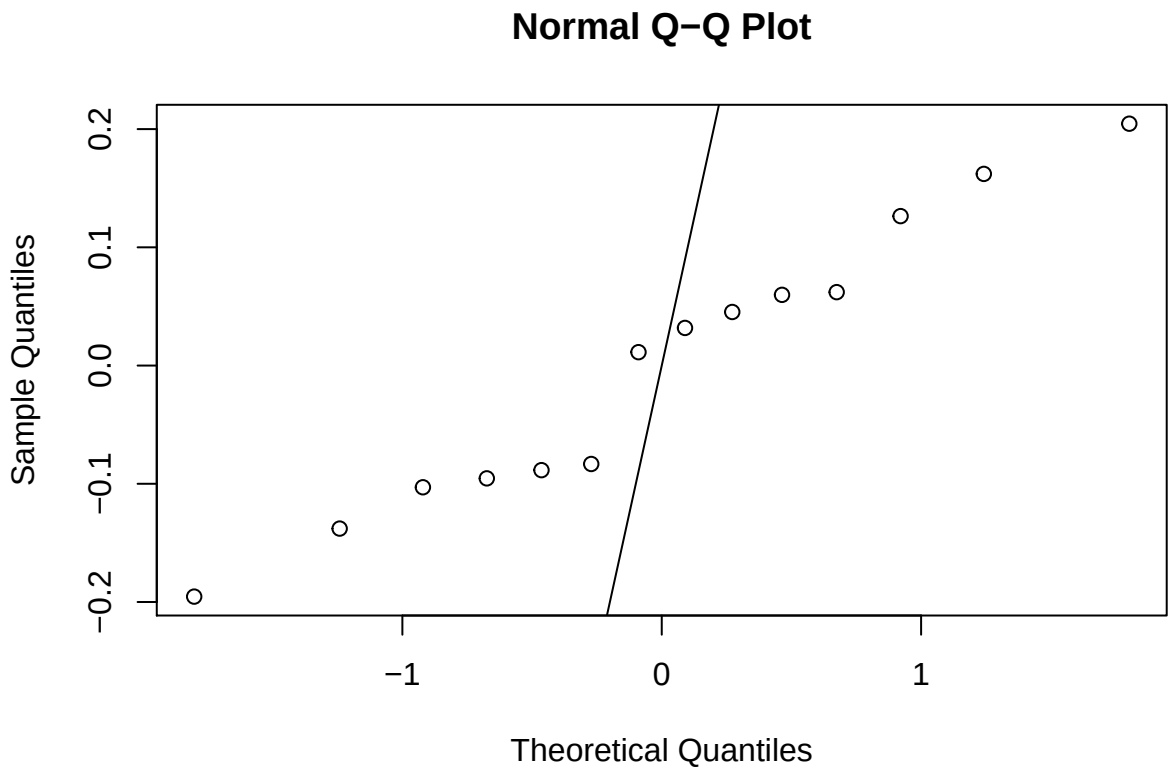
```
 names(CR.rsm2)
```

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

residuals é a variável que nos interessa, pois essa quarda o valor dos resíduos.

Para verificar a normalidade dos mesmos vamos utilizar a função *qqnorm* que compara os quantis dos dados com os da dirtribuição normal.

```
qqnorm(CR.rsm2$residuals)
 abline(0,1)
```



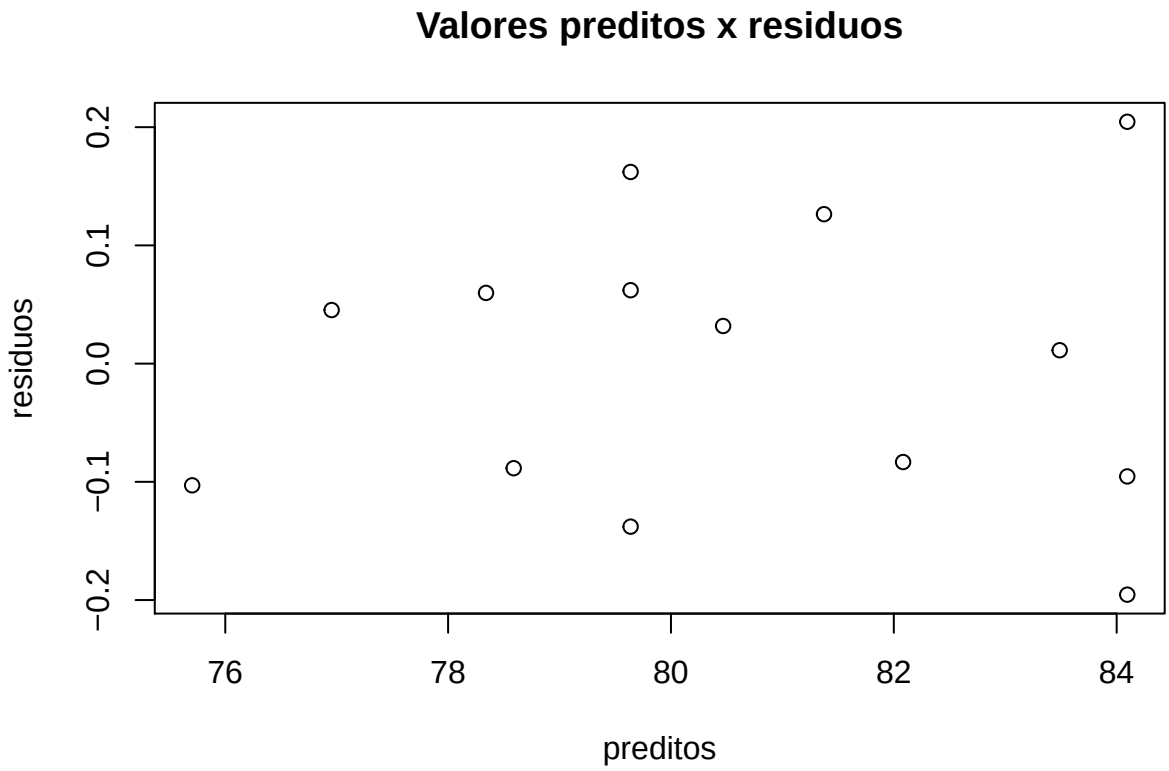
Como os pontos não estao seguindo a reta, não podemos dizer que os resíduos seguem uma distribuição normal.

3.2 Variância Constante

Para verificarmos se a variância dos resíduos é constante(homocedasticidade), vamos plotar o gráfico dos valores preditos x resíduos que estão quardados na variável CR.rsm2 da seguinte maneira.

```
residuos = CR.rsm2$residuals
preditos = CR.rsm2$fitted.values

plot(preditos,residuos, main = "Valores preditos x residuos")
```

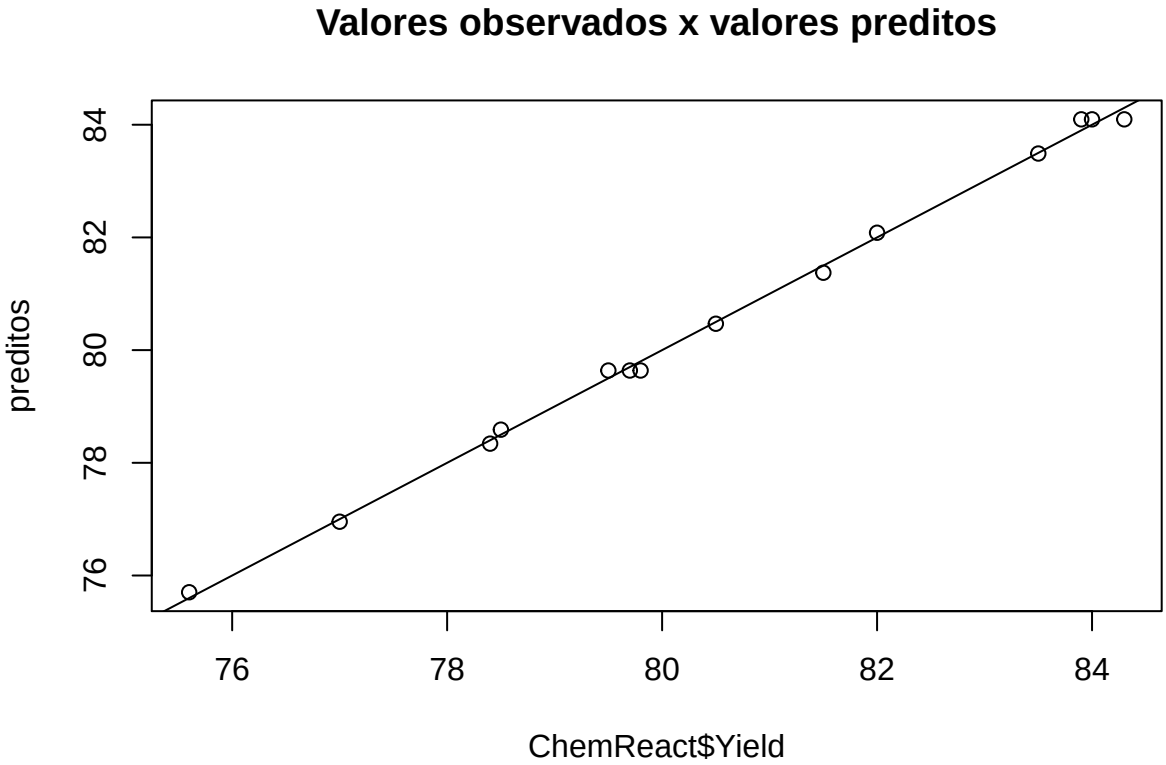


Olhando para o gráfico não há padrões aparentemente detectáveis, ou seja, a variância parece ser constante.

3.3 Ajuste do Modelo

Um modo de se verificar a qualidade do ajuste é plotar o gráfico de valores observados x valores preditos. Quanto mais próximos à reta identidade, melhor é a qualidade do ajuste.

```
 preditos = CR.rsm2$fitted.values
plot(ChemReact$Yield, preditos, main ="Valores observados x valores preditos")
abline(0,1)
```



Portanto, notamos que o modelo possui boa qualidade de ajuste, já que prevê corretamente boa parte das vezes, e quando comete erro não se distancia muito da reta identidade.

4 Comentários Finais

O conjunto de técnicas usando a metodologia de superfície de resposta são muito úteis para otimização de processos.

Primeiramente codificamos os dados (*ChemReact*, presente no pacote RSM) em R, de variáveis naturais para variáveis codificadas, e então propomos alguns modelos e verificamos qual deles se adequou melhor aos dados. Após isso, realizamos uma análise de resíduos para validar ou não nossa análise por superfície de resposta. Contudo os resíduos não possuíram normalidade mas foi um bom exemplo de como aplicar tal análise no cotidiano.

Referências bibliográficas

Comparini A., Passos G., Graziadei H., Ferreira-Silva P.,
Louzada F. **Metodologia de Superfície de Respota: Uma Introdução no software R**