响应面分析之 python 方法（二）二阶曲面分析

当实验者相对接近最优点时，通常需要一个具有弯曲性的模型来副近响应，在大多数情况下，二阶模型是合适的。

例 11.2 我们继续分析 11.1 的化学过程，仅用表 11.4 的设计不能拟合变量x_1和x_2的二阶模型。实验者决定以足够的点增大这个设计使得能拟合一个二阶模型。她在（$x_1 = 0$，x_2=±1.414）和（$x_1 = \pm 1.414$，x_2=0）处得到 4 个观测值。完整的实验列在表 11.6 中，设计显示在图 11.10 中，此设计称为中心复合设计（CCD）。在研究的第二阶段，考虑另两个响应：产品的粘度和分子量，这两个响应也列在表 11.6 中。

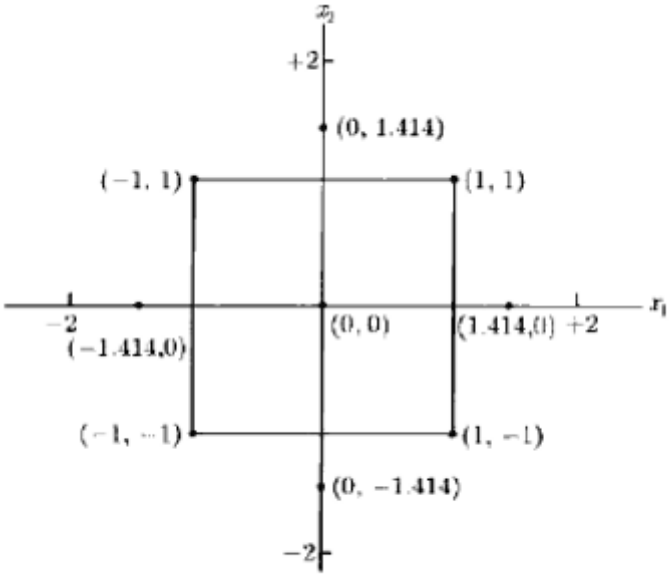


图 11.10　例 11.2 的中心复合设计

表 11.6 例 11.2 的中心复合设计

自然变量		规范变量		响应		
ξ_1	ξ_2	x_1	x_2	y1（产率）	y2（粘度）	y3（分子量）
80	170	-1	-1	76.5	62	2940
80	180	-1	1	77.0	60	3470
90	170	1	-1	78.0	66	3680
90	180	1	1	79.5	59	3890
85	175	0	0	79.9	69	3480
85	175	0	0	80.3	68	3200
85	175	0	0	80.0	70	3410
85	175	0	0	79.7	71	3290
85	175	0	0	79.8	68	3500
92.07	175	1.414	0	78.4	70	3360
77.93	175	-1.414	0	75.6	71	3020
85	182.07	0	1.414	78.5	58	3630
85	167.93	0	-1.414	77.0	57	3150

这里主要考虑拟合产率响应 y_1 的二次模型。

y_1=79.94+0.99*x[0]+0.52*x[1]+0.25* x[0]* x[1]-1.38*(x[0]**2)-1.00*(x[1]**2)

通常使用计算机软件来拟合响应曲面并画出等高线图。图 11.11 显示了由过程变量时间和温度表示的产率响应的三维响应曲面图形及对应的等高线图。考察这两张图形，可以相对容易地看出，最优点非常接近 175℉和 85 分钟反应时间，而且响应在这一点达到最大值。从等高线图可以看出，过程对反应时间的变化比对温度的变化更为敏感一些。

用（11.7）式中的通解也能求出稳定点的位置，注意到

$$b=\begin{bmatrix}0.995\\0.515\end{bmatrix} \quad B=\begin{bmatrix}-1.376 & 0.1250\\0.1250 & -1.001\end{bmatrix}$$

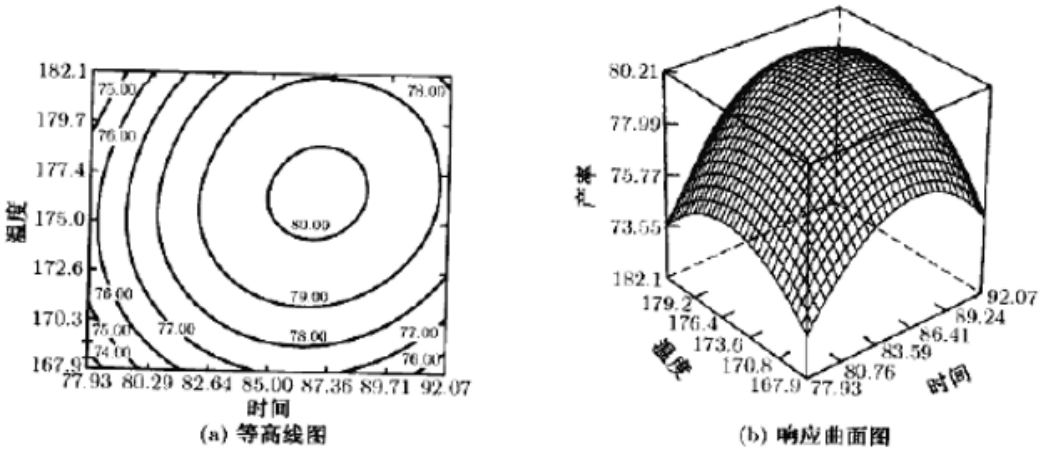
由（11.7）式，稳定点是

$$x_s=-\frac{1}{2}B^{-1}b=-\frac{1}{2}\begin{bmatrix}-0.7345 & -0.0917\\-0.0917 & -1.0096\end{bmatrix}\begin{bmatrix}0.995\\0.515\end{bmatrix}=\begin{bmatrix}0.389\\0.306\end{bmatrix}$$

即$x_{1,\,s}=0.389$，$x_{2,\,s}=0.306$。转换为自然变量，稳定点满足

$$\mathbf{0.389}=\frac{\xi_1-85}{5} \quad 和 \quad \mathbf{0.306}=\frac{\xi_2-175}{5}$$

得$\boldsymbol{\xi_1}=86.95\approx 87$ 分钟反应时间，$\boldsymbol{\xi_2}=176.53\approx 176.5$℉。这非常靠近在图 11.11 的等高线图中的目测到的稳定点。使用（11.8）式，求得在稳定点处的预测响应是$\hat{y}_s=80.21$。



(a) 等高线图　　(b) 响应曲面图

图 11.11　例 11.2 产率响应的等高线图和响应曲面图

也可以用本节描述的正则分析法来刻画响应面。首先，需要将所拟合的模型用正则形式[(11.9)式]来表示出来。特征值λ1 和 λ2是行列式方程

$$|\mathrm{B}-\lambda\mathrm{I}|=0$$

$$\begin{vmatrix}-1.376-\lambda & 0.1250\\0.1250 & -1.001-\lambda\end{vmatrix}=0$$

的根，把上面的行列式方程化简即得

$$\lambda^2+2.378\lambda+1.3639=0$$

此二次方程的根是$\lambda_1=-0.9641$ 和$\lambda_2=-1.4147$。于是，所拟合的模型的正则形式是

$$\hat{y}=80.21-0.9641\omega_1^2-1.4147\omega_2^2$$

因为λ_1和λ_2都是负的，且稳定点在探测区域内。我们的结论是，稳定点是最大值点。

在有些 RSM 问题中，必须求出正则变量$\{\omega_i\}$和设计变量$\{x_i\}$之间的关系。当过程不能在稳定点处运行时，尤其需要这样做。作为说明，设在例 11.2 中，过程不能在$\boldsymbol{\xi_1}$=87 分钟和

ξ_2=176.5℉处运行，因为这一因子组合导致成本过大。现在想从稳定点“返回”至一个较低成本的点又不至于在产率上有较大的损失。模型的正则形式显示出曲面沿ω_1方向的产率损失较小。正则形式的研究需要将（ω_1，ω_2）空间中的点变换为（x_1, x_2）空间中的点。

一般说来，变量 x 与正则变量ω之间的关系是

$$\omega = M'（x - x_s）$$

其中 M 是（k × k）正交矩阵。M 的列是与$\{\lambda_i\}$对应的标准化特征向量。也就是说，如果m_i是 M 的第 i 列，则m_i是

$$(B - \lambda_i I)m_i = 0 \qquad (11.10)$$

的解，而且$\sum_{j=1}^{k} m_{ji}^2 = 1$。

我们用例 11.2 的二阶拟合模型来说明这一方法。对λ_1=-0.9641，（11.10）式成为

$$\begin{bmatrix} -1.376 + 0.9641 & 0.1250 \\ 0.1250 & -1.001 + 0.9641 \end{bmatrix} \begin{bmatrix} m_{11} \\ m_{21} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

或

$$-0.4129m_{11} + 0.1250m_{21} = 0$$
$$0.1250m_{11} - 0.0377m_{21} = 0$$

我们要求此方程组的标准化解，即$m_{11}^2 + m_{21}^2 = 1$ 的解。此方程组没有唯一的解，所以，最方便的方法是对一个未知数指定一个任意值，解此方程组，然后将解标准化。令m_{21}^*=1，求得m_{11}^*=0.3027。要使这一解标准化，将m_{11}^*和m_{21}^*除以

$$\sqrt{（m_{11}^*）^2 + （m_{21}^*）^2} = \sqrt{0.3027^2 + 1^2} = 1.0448$$

这就得出标准化解：

$$m_{11} = \frac{m_{11}^*}{1.0448} = \frac{0.3027}{1.0448} = 0.2897$$

$$m_{21} = \frac{m_{21}^*}{1.0448} = \frac{1}{1.0448} = 0.9571$$

这就是 M 矩阵的第一列。

用$\lambda_2 = -1.4147$，重复上述步骤，求得m_{12}=-0.9574 和m_{22}=0.2888，即是 M 的第二列。于是有

$$M = \begin{bmatrix} 0.2897 & -0.9574 \\ 0.9571 & 0.2888 \end{bmatrix}$$

ω和 x 之间的关系是

$$\begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix} = \begin{bmatrix} 0.2897 & 0.9574 \\ -0.9571 & 0.2888 \end{bmatrix} \begin{bmatrix} x_1 - 0.389 \\ x_2 - 0.306 \end{bmatrix}$$

或

$$\omega_1 = 0.2897(x_1 - 0.389) + 0.9571（x_2 - 0.306）$$
$$\omega_2 = -0.957(x_1 - 0.389) + 0.2888（x_2 - 0.306）$$

如果想研究稳定点附近的响应曲面，我们就应该在（ω_1，ω_2）空间内确定合适的点，这些点上取观测值，然后用上述的关系式将这些点变换为（x_1，x_2）空间中的点，那样，就可以进行试验了。

以上内容参见《实验设计与分析》（第 6 版）。

下面提供 python 代码（文件链接为 http://www.aluoyun.cn/images/code11-3.txt）：

```
#code11-3.py
```

#本代码作者李绍安联系电话 13712566524，转载请注明出处：www.aluoyun.cn

```
from pyDOE2 import *

import statsmodels.formula.api as smf
import pandas as pd
import statsmodels.api as sm
from sklearn import linear_model
import seaborn as sns

import matplotlib.pyplot as plt

import numpy as np
from scipy.optimize import minimize


#ccdesign(n, center, alpha, face)

#其中 n 是因子数,center 是中心点,默认(4, 4)),#alpha 是“正交” (或 “o”, 默认)或 “旋转” (或“r”) ,

#face 是“外接圆” (或 “ccc”, 默认), “内切” (或 “cci”), 或 “面” (或 “ccf”).

#ccdesign(3, center=(0, 1), alpha='r', face='cci')

myccd=ccdesign(2, center=(0, 5), alpha='r')
y1=[76.5,78.0,77.0,79.5,75.6,78.4,77.0,78.5,79.9, 80.3,80.0,79.7,79.8]
y2=[62,66, 60,59,71,68,57,58,72,69,68,70,71]
y3=[2940,3680,3470,3890,3500,3360,3150,3630,3480,3200,3410,3290, 3500]
df=pd.DataFrame(myccd)
df['y1']=y1
df['y2']=y2
df['y3']=y3

model1 = smf.ols('df.y1 ~df[0] +df[1]+df[0]:df[1]+I(df[0]**2)+I(df[1]**2)', data=df).fit()
#model2 = smf.ols('df.y2 ~df[0] +df[1]+df[0]:df[1]+I(df[0]**2)+I(df[1]**2)', data=df).fit()
#model3 = smf.ols('df.y3 ~df[0] +df[1]+df[0]:df[1]+I(df[0]**2)+I(df[1]**2)', data=df).fit()
#model3 = smf.ols('df.y3 ~df[0]+df[1]',data=df).fit()

print(model1.summary2())
print(model1.params)
anovatable1=sm.stats.anova_lm(model1)
#print(model2.summary2())
#print(model2.params)
#anovatable2=sm.stats.anova_lm(model2)
#print(model3.summary2())
#print(model3.params)
#anovatable3=sm.stats.anova_lm(model3)
```

```
>>> print(model1.params)
Intercept             79.940000
df[0]                  0.994975
df[1]                  0.515165
df[0]:df[1]            0.250000
I(df[0] ** 2)         -1.376250
I(df[1] ** 2)         -1.001250
dtype: float64
```

```
# 定义目标函数和约束条件
def objective(x):
    return -(79.94+0.99*x[0]+0.52*x[1]+0.25* x[0]* x[1]-1.38*(x[0]**2)-1.00*(x[1]**2))
def constraint1(x):
    return -x[0]**2+1
def constraint2(x):
    return   -x[1]**2+1
# 定义初始点
x0 = np.array((-0.9, -0.9))
# 使用 SLSQP 算法求解非线性规划问题
solution = minimize(objective, x0, method='SLSQP', constraints=[{'fun': constraint1, 'type': 'ineq'},
                                                                 {'fun': constraint2, 'type': 'ineq'} ])
print(solution)
```

```
>>> print(solution)
 message: Optimization terminated successfully
 success: True
  status: 0
     fun: -80.21154411360513
       x: [ 3.866e-01    3.083e-01]
     nit: 3
     jac: [ 0.000e+00    9.537e-07]
    nfev: 11
    njev: 3
```

即最大值为80.21，在（0.387，0.308）处。