SAS Examples for Response Surface Regression and Nonlinear Regression

December 5, 2000

1 Response Surface Regression

SAS's PROC RSREG performs response surface regression, assuming a local quadratic fit and computing parameter estimates, the location of any stationary point, and predictions. The additional RIDGE feature may be used to determine the maximum or minimum at a sequence of fixed distances from the starting point, this allowing the user to envision the direction of steepest ascent or descent as one proceeds away from the starting point.

Here is an example, applied to the data of Table 7.1:

```
options ls=64 ps=58;
data rs1;
input x1 x2 y;
datalines;
 0  0  0.750
 0  1  3.367
 0 -1  0.535
 1  0  0.896
 1  1  3.039
 1 -1  0.798
-1  0  3.154
-1  1  5.010
-1 -1  2.872
;
proc rsreg data=rs1 out=p;
model y=x1 x2 /predict ;
ridge min;
run;
proc print data=p;
var x1 x2 y;
run;
```

In this example, the two predictor variables are denoted 'x1' and 'x2', and the response variable is denoted 'y'. The procedure allows as many response variables as are desired, but always fits a quadratic model in the response variables. The option 'predict' creates predicted values. The additional command 'ridge min' plots the ridge on which the minimum is achieved as one moves away from the starting point (0,0).

The output from this program is as follows:

```
                    The RSREG Procedure

       Coding Coefficients for the Independent Variables

           Factor    Subtracted off      Divided by

           x1                     0        1.000000
           x2                     0        1.000000


               Response Surface for Variable y

           Response Mean                   2.269000
           Root MSE                        0.255013
           R-Square                          0.9902
           Coefficient of Variation         11.2390


                         Type I Sum
Regression        DF    of Squares  R-Square  F Value  Pr > F

Linear             2     15.287722    0.7713   117.54  0.0014
Quadratic          2      4.336037    0.2188    33.34  0.0089
Crossproduct       1      0.002652    0.0001     0.04  0.8529
Total Model        5     19.626411    0.9902    60.36  0.0033


                                    Sum of
    Residual            DF         Squares     Mean Square

    Total Error          3        0.195095        0.065032


                                   Standard
 Parameter  DF      Estimate          Error  t Value  Pr > |t|
```

Intercept	1	0.881667	0.190076	4.64	0.0189
x1	1	-1.050500	0.104109	-10.09	0.0021
x2	1	1.201833	0.104109	11.54	0.0014
x1*x1	1	1.077500	0.180322	5.98	0.0094
x2*x1	1	0.025750	0.127507	0.20	0.8529
x2*x2	1	1.003500	0.180322	5.57	0.0114

Parameter	Parameter Estimate from Coded Data
Intercept	0.881667
x1	-1.050500
x2	1.201833
x1*x1	1.077500
x2*x1	0.025750
x2*x2	1.003500

Factor	DF	Sum of Squares	Mean Square	F Value	Pr > F
x1	3	8.945966	2.981989	45.85	0.0053
x2	3	10.683097	3.561032	54.76	0.0041

Canonical Analysis of Response Surface Based on Coded Data

Factor	Critical Value Coded	Uncoded
x1	0.494702	0.494702
x2	-0.605168	-0.605168

Predicted value at stationary point: 0.258169

Eigenvalues	Eigenvectors x1	x2
1.079676	0.986016	0.166653
1.001324	-0.166653	0.986016

Stationary point is a minimum.

```
Estimated Ridge of Minimum Response for Variable y

 Coded     Estimated      Standard     Uncoded Factor Values
Radius      Response         Error            x1            x2

   0.0      0.881667      0.190076             0             0
   0.1      0.732270      0.189224      0.065482     -0.075578
   0.2      0.603320      0.186711      0.130311     -0.151721
   0.3      0.494808      0.182673      0.194490     -0.228415
   0.4      0.406725      0.177356      0.258027     -0.305651
   0.5      0.339062      0.171154      0.320926     -0.383414
   0.6      0.291811      0.164652      0.383194     -0.461695
   0.7      0.264964      0.158694      0.444837     -0.540482
   0.8      0.258513      0.154412      0.505861     -0.619762
   0.9      0.272449      0.153203      0.566274     -0.699524
   1.0      0.306765      0.156539      0.626082     -0.779757

              Obs    x1    x2       y

                1     0     0    0.88167
                2     0     1    3.08700
                3     0    -1    0.68333
                4     1     0    0.90867
                5     1     1    3.13975
                6     1    -1    0.68458
                7    -1     0    3.00967
                8    -1     1    5.18925
                9    -1    -1    2.83708
```

The first part of the output calculates an analysis of variance table, followed by parameter estimates and standard errors.

The ‘canonical analysis’ finds the stationary point of the fitted curve, the predicted response at the stationary point, and determines whether this is a local maximum, a local minimum or a saddlepoint. The eigenvalues are those of the Hessian matrix of the fitted surface at the estimated stationary point; in this case both eigenvalues are positive, so the stationary point is a local minimum. If both eigenvalues were negative, it would be a local maximum, and if one was positive and the other negative, it would be a saddlepoint.

The ‘estimated ridge’ shows the minimum on fixed radii as one moves away from the starting point. The minimum value over those tabulated is achieved at radius 0.8.

The final part of the output shows the predicted values at the original data points. It would be possible to calculate the predicted responses for a much denser grid of (x1,x2) values and to use the results to obtain a contour plot of the estimated response.

2 Nonlinear Regression: The Michaelis-Menten Model

Recall that the Michaelis-Menten model is defined by the equation

$$y_i = \frac{\theta_1 x_i}{\theta_2 + x_i} + \epsilon_i, \tag{1}$$

where θ_1 and θ_2 are unknown parameters and the ϵ_i terms are errors satisfying the usual assumptions (independent $N(0, \sigma^2)$ for common unknown σ^2).

We also noted in Section 7.3 that a plot of $1/y_i$ against $1/x_i$ could be expected to be roughly linear, though we recommended against this method as a means of fitting the model.

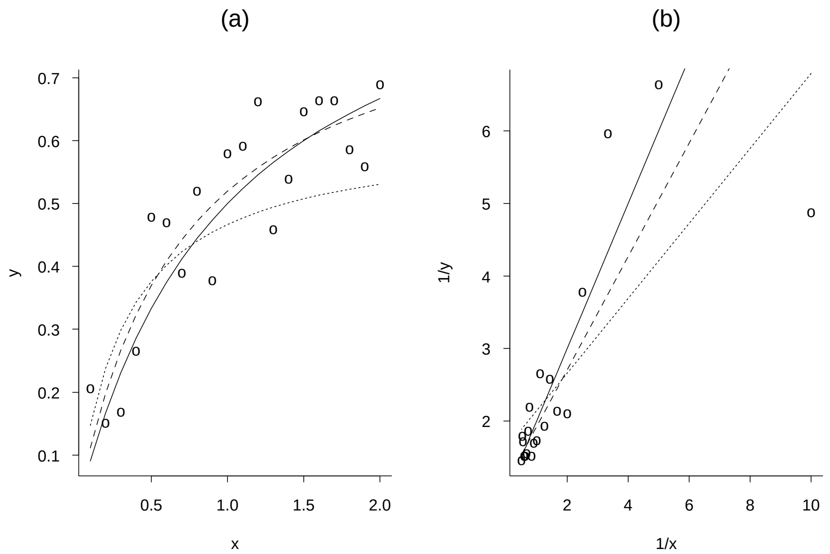

Figure 1. Simulated data set for Michaelis-Menten example, (a) plotted on original scale, (b) plotted on the $1/y$ *vs.* $1/x$ scale which linearizes the fit. The solid curve (or line in (b)) is the true curve from which the data were generated; the dotted curve is obtained from the least squares fit in (b); the dashed curve is obtained by nonlinear regression in (a).

As an example of this, a simulation has been performed with $x_i = 0.1, 0.2, ..., 2$, $\theta_1 = \theta_2 = 1$, $\sigma = 0.1$. Fig. 1(a) shows a plot of the raw data together with the true model curve $y = x/(1 + x)$ (solid curve), the fitted curve obtained after regressing $1/y_i$ on $1/x_i$ (dotted curve), and the fitted curve after using the nonlinear regression method to be described (dashed curve). Fig. 1(b) shows the same three curves plotted on a $1/y$ *vs.* $1/x$ scale. It can be seen that one apparent outlier on the plot (the extreme right-hand point in Fig. 1(b)) has significantly distorted the linear fit (the dotted line), but the nonlinear regression fit (the dashed line) is much closer to the true regression line.

Now to the details of how to obtain the nonlinear fit. Here is the SAS program that was used in this example:

```
options ls=64 ps=58;
data mm1;
input x y;
datalines;
 0.100000 0.205445
 0.200000 0.150658
 0.3000000 0.1677381
 0.4000000 0.2648728
 0.5000000 0.4777572
 0.6000000 0.4688204
 0.7000000 0.3891154
 0.8000000 0.5192841
 0.9000000 0.3769132
 1.0000000 0.5790269
 1.1000000 0.5909103
 1.200000 0.661661
 1.3000000 0.4576434
 1.4000000 0.5379632
 1.5000000 0.6458155
 1.6000000 0.6627638
 1.700000 0.662643
 1.8000000 0.5849613
 1.9000000 0.5581459
 2.0000000 0.6888268
;
proc nlin method=gauss;
parms th1=.6148 th2=.3176;
model y=th1*x/(th2+x);
der.th1=x/(th2+x);
der.th2=-th1*x/(th2+x)**2;
output out=mm2 p=yhat r=resid;
run;
proc print data=mm2;
var x yhat resid;
run;
```

Lines 5–24 input the data points, which are the same as shown in the plots of Fig. 1. Lines 26–32 define the call to 'proc nlin'. Features of this are:

1. 'method=gauss' specifies the method of optimization — in this case, the Gauss-Newton method. Other options are 'method=gradient' for a steepest descent method, 'method=marquardt' for the Levenberg-Marquardt algorithm, 'method=newton' for Newton-Raphson, and 'method=dud' for a derivative-free algorithm (secant method or method of false position).

2. The 'parms' statement specifies the initial values of the two parameters. In this case, these are the values obtained by the regression of $1/y_i$ on $1/x_i$ (the dotted curve of Fig. 1(a)).

3. The 'model' statement specifies the response as a function of parameters 'th1' and 'th2'.

4. The 'der.th1' and 'der.th2' specify the first-order derivatives of the response with respect to the two parameters. We need this because the Gauss-Newton method requires first-order derivatives, as do all the above-mentioned methods except 'dud'. For 'method=newton', we would have to specify second-order derivatives as well ('der.th1.th1', etc.).

5. The 'output' statement used here specifies that predicted values and residuals will be saved in a data set 'mm2', and the subsequent 'proc print' statement prints these values.

The output from this program is as follows:

```
                     The NLIN Procedure
                       Iterative Phase
                    Dependent Variable y
                    Method: Gauss-Newton

                                                 Sum of
          Iter          th1          th2        Squares

             0       0.6148       0.3176         0.2139
             1       0.8240       0.6031         0.1113
             2       0.8743       0.6833         0.1091
             3       0.8753       0.6845         0.1091
             4       0.8753       0.6845         0.1091

NOTE: Convergence criterion met.

                     Estimation Summary

            Method                    Gauss-Newton
```

Iterations	4
R	2.04E-6
PPC(th2)	2.731E-6
RPC(th2)	0.00007
Object	2.613E-9
Objective	0.109054
Observations Read	20
Observations Used	20
Observations Missing	0

NOTE: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Regression	2	5.1100	2.5550	421.72	<.0001
Residual	18	0.1091	0.00606		
Uncorrected Total	20	5.2191			
Corrected Total	19	0.5620			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
th1	0.8753	0.1048	0.6550	1.0955
th2	0.6845	0.2160	0.2307	1.1382

Approximate Correlation Matrix

	th1	th2
th1	1.0000000	0.9577979
th2	0.9577979	1.0000000

Obs	x	YHAT	RESID
1	0.1	0.11158	0.09387
2	0.2	0.19792	-0.04726
3	0.3	0.26672	-0.09899
4	0.4	0.32284	-0.05797
5	0.5	0.36948	0.10828
6	0.6	0.40885	0.05997
7	0.7	0.44254	-0.05343
8	0.8	0.47169	0.04759
9	0.9	0.49716	-0.12025

10	1.0	0.51961	0.05942
11	1.1	0.53954	0.05137
12	1.2	0.55735	0.10431
13	1.3	0.57337	-0.11573
14	1.4	0.58786	-0.04989
15	1.5	0.60101	0.04480
16	1.6	0.61302	0.04975
17	1.7	0.62402	0.03863
18	1.8	0.63413	-0.04917
19	1.9	0.64346	-0.08531
20	2.0	0.65209	0.03673

The most important feature to note here is that the algorithm converged in four iterations to the nonlinear least squares estimates ($\widehat{\theta}_1 = 0.8753$, $\widehat{\theta}_2 = 0.6845$, with standard errors .1048, .2160 respectively) and these are the estimates plotted as the dashed curves in Fig. 1.

3 Nonlinear Least Squares Applied to Running Times Data

Table 1 shows best performances by year for the men's mile race, 1913–1997. Fig. 2 shows the same data plotted, with a least squares straight line (solid line). However, Fig. 2 also shows a nonlinear fit to the same data, which we describe below.

Here is some S-PLUS code that was used to explore this example:

```
mmile<-read.table('mmile1.txt',header=T,row.names=NULL)
attach(mmile)
postscript(file='mmile.ps',horizo=F)
par(mfrow=c(2,1),bty='l',las=1,oma=c(0,0,0,0))
nreg<-lm(mmtime~yr1)
print(summary(nreg)$sigma)
plot(yr1,mmtime,xlab='Year',ylab='Mile time')
lines(yr1,predict(nreg))
for(i in 1:10){
x2<-exp(-0.001*i*(yr1-1955))
nreg<-lm(mmtime~x2)
print(c(i,summary(nreg)$sigma))}
for(i in 25:35){
x2<-exp(-0.0001*i*(yr1-1955))
nreg<-lm(mmtime~x2)
print(c(i,summary(nreg)$sigma))}
x2<-exp(-.0029*(yr1-1955)
```

Year	Time	Year	Time	Year	Time	Year	Time
1913	254.40	1935	246.80	1956	238.60	1977	232.00
1914	254.40	1936	246.80	1957	237.20	1978	232.50
1915	252.60	1937	246.40	1958	234.50	1979	228.95
1916	252.60	1938	247.20	1959	236.50	1980	228.80
1917	252.60	1939	247.40	1960	237.00	1981	227.33
1918	252.60	1940	248.30	1961	237.60	1982	227.69
1919	252.60	1941	248.60	1962	234.40	1983	229.21
1920	252.60	1942	244.60	1963	234.90	1984	229.54
1921	253.80	1943	242.60	1964	234.10	1985	226.32
1922	257.00	1944	241.60	1965	233.60	1986	228.31
1923	250.40	1945	241.40	1966	231.30	1987	226.76
1924	257.80	1946	246.60	1967	231.10	1988	228.85
1925	255.20	1947	247.00	1968	233.80	1989	229.90
1926	251.90	1948	248.80	1969	235.90	1990	229.31
1927	256.80	1949	245.40	1970	236.30	1991	229.12
1928	251.00	1950	246.20	1971	234.40	1992	228.80
1929	250.40	1951	241.40	1972	232.80	1993	224.39
1930	250.40	1952	242.10	1973	232.00	1994	228.67
1931	249.20	1953	242.00	1974	233.20	1995	225.19
1932	249.20	1954	237.90	1975	229.40	1996	228.15
1933	247.60	1955	239.00	1976	233.10	1997	224.90
1934	246.80						

Table 1: Table of best performances in men's mile race, 1913–1997

```
nreg<-lm(mmtime~x2)
lines(yr1,predict(nreg),lty=3)
dev.off()
```

In this case a data frame 'mmile' is used to contain the year variable 'yr1' and the best time 'mmtime'. The first 'lm' statement simply fits a linear regression of 'mmtime' on 'yr1', which is added to the plot.

The alternative, non-linear, model in this instance is

$$t_i = \theta_1 + \theta_2 \exp\{-\theta_3(y_i - 1955)\} + \epsilon_i, \qquad (2)$$

where y_i denotes year i and t_i denotes the running time in year i. The model (2) has some attractive features because, when $\theta_3 > 0$, the parameter θ_1 represents an 'ultimate limit', which some people have claimed to be able to estimate from data of this nature. The value y_i is centered about 1955 (the mean year in this data set) to improve numerical stability.

As an example of the output, here is the sequence of 'print' statements produced by the above program:

```
[1] 2.403803

[1] 1.000000 2.391518
[1] 2.000000 2.384372
[1] 3.000000 2.382413
[1] 4.000000 2.385646
[1] 5.000000 2.394031
[1] 6.000000 2.407484
[1] 7.000000 2.425881
[1] 8.000000 2.449061
[1] 9.000000 2.476832
[1] 10.000000  2.508973

[1] 25.000000  2.382742
[1] 26.000000  2.382572
[1] 27.000000  2.382454
[1] 28.000000  2.382389
[1] 29.000000  2.382375
[1] 30.000000  2.382413
[1] 31.000000  2.382503
[1] 32.000000  2.382645
[1] 33.000000  2.382838
[1] 34.000000  2.383084
[1] 35.000000  2.383382
```

The first line here shows the residual estimate of σ under the least squares model (2.403803). The next 10 lines show the corresponding values for $\theta_3 =$

.001, .002, ... .01, achieving a minimum among these values when $\theta_3 = .003$. This result promted the insertion of the additional block of code (beginning 'for(i in 25:35)') which explores values of θ_3 from .0025 to .0035. However, even the minimum value (2.382375 at $\theta_3 = .0029$) is not significantly different from the linear fit; the corresponding values for the residual sums of squares are 479.5963 ($=83 \times 2.403803^2$) and 471.084, and treating θ_3 as an additional parameter that has been estimated, the F statistic is $82 \times (479.5963 - 471.084)/471.084 = 1.455$, not significant as an $F_{1,82}$ variable (p value 0.23).

Now let us see how to fit the model (2) using PROC NLIN. Here is the SAS code:

```
options ls=64 ps=58;
data mmile1;
infile 'mmile1.txt';
input yr time;
yr1=yr-1955;
;
proc nlin method=gauss;
parms th1=130.9 th2=108.77 th3=0.01;
model time=th1+th2*exp(-th3*yr1);
der.th1=1;
der.th2=exp(-th3*yr1);
der.th3=-th2*yr1*exp(-th3*yr1);
output out=mmile2 p=yhat r=resid;
run;
proc print data=mmile2;
var year yhat resid;
run;
```

Here is the output from the SAS code:

```
                    The NLIN Procedure
                     Iterative Phase
                  Dependent Variable time
                   Method: Gauss-Newton

                                                    Sum of
      Iter          th1          th2          th3      Squares

         0        130.9        108.8       0.0100      28771.0
         1        170.5      69.3603      0.00717       1318.9
         2        150.1      89.7117      0.00342        754.1
         3        108.0        131.8      0.00265        525.6
         4        110.1        129.6      0.00294        471.1
         5        106.9        132.8      0.00287        471.1
```

```
          6       107.3       132.4     0.00288       471.1
          7       107.3       132.5     0.00288       471.1

NOTE: Convergence criterion met.

                       Estimation Summary

              Method                      Gauss-Newton
              Iterations                             7
              R                               9.732E-6
              PPC(th1)                        0.000095
              RPC(th1)                        0.000657
              Object                          5.366E-9
              Objective                       471.0833
              Observations Read                     85
              Observations Used                     85
              Observations Missing                   0

                                 Sum of       Mean               Approx
Source                   DF     Squares     Square    F Value    Pr > F

Regression                3     4906529    1635510     649.34    <.0001
Residual                 82       471.1     5.7449
Uncorrected Total        85     4907000

Corrected Total          84      7931.9

                                   Approx        Approximate 95%
  Parameter      Estimate       Std Error       Confidence Limits

  th1               107.3           116.9      -125.3        339.8
  th2               132.5           117.2      -100.6        365.6
  th3             0.00288         0.00253    -0.00216      0.00792

                    Approximate Correlation Matrix
                          th1             th2             th3

      th1       1.0000000      -0.9999975       0.9995004
      th2      -0.9999975       1.0000000      -0.9995078
      th3       0.9995004      -0.9995078       1.0000000
```

This shows that the optimum value of θ_3 is .00288. The corresponding estimate of θ_1 is 107.3, with an estimated standard error (by the delta method) of 107.3 (i.e. a predicted "best possible time" of 1 minute, 47 seconds) with a

95% “confidence interval” of (–125.3,339.8). In this case, the final result of the nonlinear regression exercise is somewhat meaningless.

However, as an exercise, the reader is invited to consider the corresponding table of women’s times since 1966 (Table 2), which is plotted without any fitted lines or curves in Fig. 3. Is the existence of a “best possible time” more plausible in this case?

Year	Time	Year	Time	Year	Time	Year	Time
1966	284.20	1974	269.50	1982	257.44	1990	258.13
1967	277.00	1975	269.50	1983	261.65	1991	257.00
1968	282.20	1976	269.50	1984	255.80	1992	261.30
1969	276.80	1977	263.80	1985	256.71	1993	261.69
1970	276.80	1978	266.90	1986	258.25	1994	257.25
1971	275.40	1979	262.09	1987	264.05	1995	264.13
1972	275.30	1980	261.68	1988	261.25	1996	252.56
1973	269.50	1981	260.89	1989	255.61	1997	264.53

Table 2: Table of best performances in women’s mile race, 1966–1997

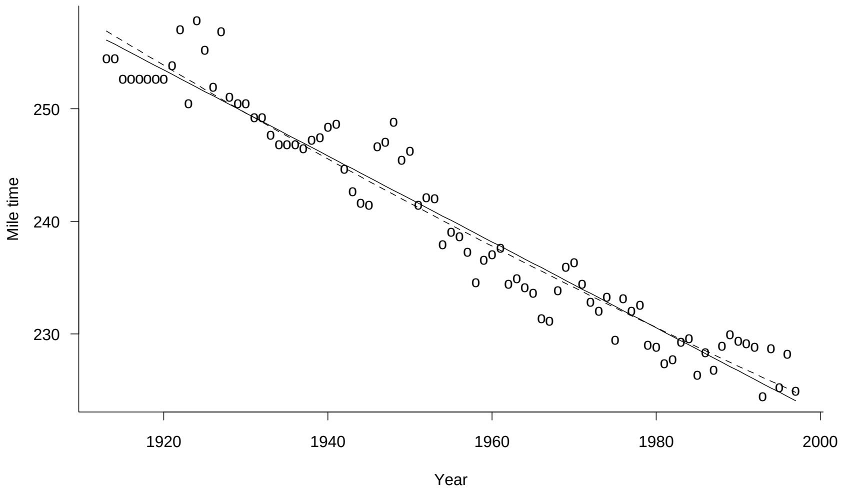

Figure 2. Best performance by year in men's mile race from 1913 to 1997, with least squares straight line fit (solid line), and the nonlinear curve fitted in the test (dashed curve).

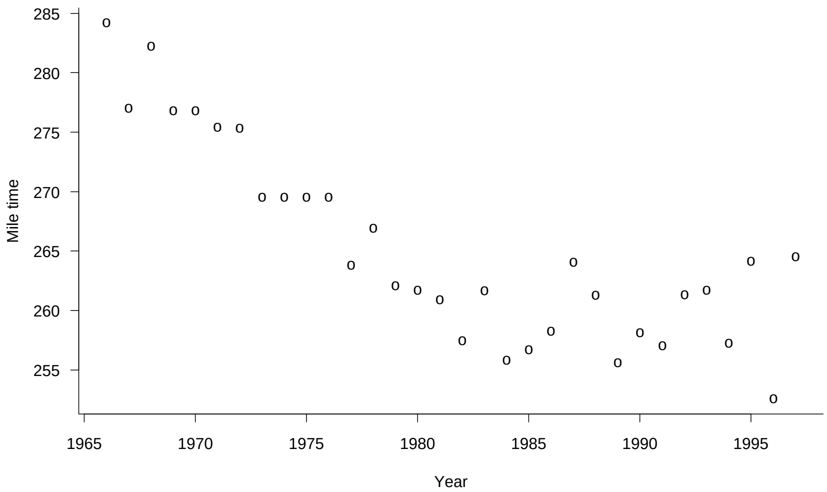

Figure 3. Best performance by year in women's mile race from 1966 to 1997.