Paper 249-26

Solving non-standard optimization problems using SAS/IML software with application to growth modeling and optimal design theory

Inna Perevozskaya, University of Maryland Baltimore County,

ABSTRACT

It is often necessary in applied statistical research to perform direct numerical optimization because the problem can not be transformed to be solved by standard procedures implemented in the statistical packages.

One example of such a problem comes from the field of growth modeling: The Polynomial Gompertz model is known to provide a good fit to individual growth curves and growth percentiles. However, this non-linear regression model cannot be fitted by PROC NLIN because of the presence of an integral in expression of the Polynomial Gompertz function. Therefore, direct optimization is necessary to obtain the least squares estimates of the model parameters.

Another example is provided by optimal design theory: in some applications, such as Phase I cancer studies, it is extremely important to implement a design that provides maximal information from a small sample, so that a minimal number of patients is exposed to the potentially toxic experimental drug. The formal optimality criteria based on Fisher's information matrix can be introduced to derive a design providing the most precise estimates for a given sample size. In many practical cases this leads to a highly non-linear optimization problem with constraints.

Both problems can be successfully solved using the advanced numerical optimization package in SAS/IML. This paper gives an overview of the package's main features and provides several examples of how optimization methods can be used to solve two very different in nature optimization problems. Some computational aspects of the problem are also discussed.

INTRODUCTION

In this section we introduce the two problems that are used to illustrate an application of SAS/IML numerical optimization routines. These problems arise from different areas of statistics: one is used for fitting non-linear curves to the growth data and the other is very helpful in planning an efficient experiment in clinical trials. We will refer to the former as the *Polynomial Gompertz growth model* and to the latter as the *Optimal Design for the Proportional Odds model*. Although these two problems are very different in nature they have a common property element: they can be solved by applying very similar non-linear optimization techniques. We describe the Proportional Odds model in detail and give only a brief overview of the Polynomial Gompertz model since it was already discussed in (Perevozskaya I., Kuznetsova O.M. (2000)).

POLYNOMIAL GOMPERTZ GROWTH MODEL

The patterns of human growth have been studied quite extensively. Besides the obvious fact that height increases with age until the final adult height is reached, it has been noticed that the growth velocity is high at birth, rapidly decelerates in infancy, slightly declines during the long period of juvenile growth, shoots up in adolescence and declines to zero while the growth curve approaches the final adult height. The parametric modeling of growth allows to summarize this complicated process with a reasonable number of parameters and facilitate many tasks in applications such as clinical trials, epidemiology and pediatrics. For instance, in some clinical studies the height, length, weight and head circumference individual curves are often plotted against the population percentile curves to detect the growth faltering in treatment groups. If the analytic expressions for percentile curves are available, this task is greatly simplified. (Perevozskaya I., Kuznetsova O.M. (2000))

The Polynomial Gompertz model was one of the many models developed for this purpose. Although it has a very simple and natural foundation, it has not been studied extensively until recently due to software limitations. The model is based on the fundamental concept of relative growth rate (RGR), which is analogous to the concept of a hazard function in survival analysis. It is defined as the ratio of the growth rate (velocity) to the achieved growth:

$$r(x) = \frac{1}{y(x)}\frac{dy(x)}{dx} = \frac{d\log y(x)}{dx},$$

where y represents one of the possible response variables such as height, weight, head circumference *etc*. and x denotes age. Although the RGR itself is a complicated function of age, the pattern of its logarithm is simpler: it can be modeled adequately by a polynomial:

$$\log r(x) = \sum_{i=0}^{K} b_i x^i \qquad (1)$$

Then the growth model can be written as :

$$y = f(x) = A_0 \exp\left(\int_0^x \exp\left(\sum_{i=0}^{K} b_i u^i \right) du \right) \qquad (2)$$

This general form involving a polynomial of degree K allows a lot of flexibility for the model to be applied to different age intervals. For example, K=5 would be good for modeling a whole life span of growth, and for K=1 we obtain the function known as Gompertz function, which is very useful in modeling fetal and early infancy human growth.

Because of the presence of an integral in the expression of the Polynomial Gompertz growth function, the SAS procedure NLIN cannot be used to obtain the ordinary least squares estimates of the parameters b_0 , …,b_K. Instead, it is necessary to directly minimize the function

$$\sum_{j=1}^{n} \left(y_j - f(x_j) \right)^2 \qquad (3)$$

of K+2 variables (A_0, b_0,…, b_K), where x_j and y_j , j=1,…, n represent the individual's age and height measurements respectively.

An easy alternative approach would be to use (1) rather than (2) as a model equation and to fit log *r(x) approximated* from the subject's data to the polynomial using SAS PROC NLIN. (We will refer to this approach as RGR fit, as opposed to fitting the Polynomial Gompertz curve using equation (2)). However, this easy approach is not precise enough. We fitted two curves to the same data set using two different methods. The results are summarized in Figure 1 which shows the plots of height, growth

velocity and RGR for both methods. The three graphs on the left correspond to direct minimization of (3). In this case it is the height curve itself that was fitted. The other three graphs show the relative velocity curve fit, which the height plot was derived from. It can easily be seen that the RGR fitted curve underestimates or overestimates the data systematically. Therefore, it cannot be considered a good solution. However it can be very useful for determining the starting point for the optimization algorithm, which can be crucial for a good convergence.

OPTIMAL DESIGN FOR THE PROPORTIONAL ODDS MODEL

Phase I clinical trial are typically small, uncontrolled sequential studies of human subjects designed to determine the maximum tolerated dose (MTD) of an experimental drug. An accurate determination of the MTD is particularly important in cancer studies because of the severity of the side effects of cytotoxic drugs. If the MTD value is underestimated, this may result in low efficacy of the treatment drug. On the other hand, overestimating the MTD would expose a large number of patients to potentially dangerous dose levels when the MTD is passed for further testing in Phase II studies.

Consequently, it is necessary to design the trial so that maximum information is obtained from minimal number of patients. Currently there is no unique philosophy on how the MTD is defined. Some authors believe that it can be identified directly from the patient data. This is the approach most commonly used in practice (Babb et all.,1998). Others treat it as a quantile of a monotonic dose response curve, which must be estimated. We will follow the second approach and use a formal optimality criteria to select the most efficient design.

Suppose we are planning an experiment where N patients are assigned to different dose levels $\{d_1,\dots d_n\}$ of a certain drug. Their responses $Y_1,..,Y_N$ are observed and classified into several ordered categories, each corresponding to some degree of toxicity (we will use "no toxicity", "moderate toxicity", "severe toxicity" and "death" in our example). The investigators are often interested in efficient estimation of the dose levels corresponding to prespecified probabilities for each type of toxicity. For a given N, the "distribution" of patients across the dose levels (as well as the choice of dose levels itself) can affect the precision of estimation. Therefore, it is desirable to find a design which results in the smallest possible variance of the estimators

STATISTICAL DETAILS AND FORMAL OPTIMALITY CRITERIA

A design ξ is formally defined as a set of dose levels $\{d_1,\dots d_n\}$ and weights (proportions of patients assigned to them):

$$w_i = N_i / N, i = 1,\dots,n \ ,$$

where N_i represents the number of patients assigned to the dose d_i. The restrictive assumption on weights-the ratio of the integer numbers-can be replaced by $0 \le w_i \le 1$, $i=1,\dots n$, and $\sum w_i = 1$ to simplify the problem. If we define R_j, j=1,…4 as the number of toxicities of type j at the dose level d, then the vector $(R_1,\dots,R_4)$ has a multinomial distribution with parameters $(N;P_1, \dots P_4)$, where N is the total number of patients in the study. The Proportional Odds model is formulated in terms of cumulative response probabilities:

$$\gamma_j = P(Y \ge j) = P_j + P_{j+1} + \dots + P_4, \ j = 2,\dots,4$$

Since $\sum Pj=1,\ j=1,\dots 4$, the cumulative probabilities are completely determined by only three equations:

$$\log \frac{\gamma_j}{1-\gamma_j} = \frac{d-\alpha_j}{\beta}, \ j = 2,\dots 4$$

This model is a particular case of the general class of multinomial logistic models described in Atkinson and Zocchi (1999). It is referred to there as "the cumulative logits model of Agresti".

The quantiles of interest μ_2, μ_3, μ_4 corresponding to the three parametric curves are defined as $\mu_j = F^{-1}(\Gamma_j)$, j=2,…4, where F is the logistic function and Γ_2, Γ_3, Γ_4 are some probabilities of interest selected before initiation of the experiment. In our examples we use

$$\Gamma_2 = 0.6 = P(mild\ toxoicity),$$
$$\Gamma_3 = 0.4 = P(severe\ toxoicity),$$
$$\Gamma_3 = 0.3 = P(death).$$

The relationship between the quantiles $\boldsymbol{\mu}=(\mu_2,\mu_3,\mu_4)^T$ and the original parameters $\boldsymbol{\theta}=(\alpha_2,\alpha_3,\alpha_4,\beta)^T$ is given by $\boldsymbol{\mu}=C^T\boldsymbol{\theta}$, where C is a matrix depending on $\Gamma_2,\Gamma_3,\Gamma_4$. It is known that the asymptotic variance–covariance matrix of the estimate of $\boldsymbol{\mu}$ is proportional to $C^TM^{-1}C$. Therefore, it is desirable to minimize this product in some sense.The most common approach is to minimize $\det[C^TM^{-1}C]$ (D_A-optimality criterion) or $\mathrm{tr}[C^TM^{-1}C]$ (the "omnibus" criterion). The particular case when C is an identity matrix corresponds to the criterion known as D-optimality. This choice of C would be preferable if the experimenter is interested in efficient estimation of the set $\boldsymbol{\theta}=(\alpha_2,\alpha_3,\alpha_4,\beta)$ rather than $\boldsymbol{\mu}=(\mu_2,\mu_3,\mu_4)^T$. Finally, the optimal design problem is formulated as minimizing the determinant or trace of $C^TM^{-1}(\xi)C$ where $\xi = \{d_1,\dots,d_n;w_1,\dots w_{n-1}\}$. This problem needs to be solved by applying the techniques of non-linear minimization.

SAS/IML NUMERICAL OPTIMIZATION PACKAGE OVERVIEW

Both problems (the PG and the PO) require minimizing a non-linear function of several variables subject to linear and boundary constraints. The tools for solving such problems are currently available as built-in functions is in SAS/IML software. An important feature of the package is that the objective function (the function to be minimized) can be specified as a separate module, which is essential for our problems: both have very complicated objective functions that are impossible to write in a closed-form expression. Moreover, a single evaluation of the objective function requires calls to other numerical routines such as numerical integration (for the PG model) or matrix inversion and multiplication (for the PO model). The availability of these methods as built-in functions as well as the possibility of user-defined modules makes SAS/IML an appealing software choice for these two problems. SAS/IML offers a variety of non-linear optimization functions suitable for solving different types of problems. . Another important advantage is that all the functions have very similar syntax, which makes it very easy to try the performance of different methods on the same problem: all modifications to the code would be as simple as changing the function name.

The choice of an algorithm depends on the type of a particular problem. For the PG model we have selected the Levenberg-Marquardt least squares method, since it performs better than other more general methods for this type of problems. For the Proportional Odds model the choice was the Quasi-Newton algorithm because of its rapid convergence and relatively low computational cost. However, this method requires a good starting point to achieve this rapid convergence. In the cases when the convergence failed due to the poor initial guess, we found it useful to apply the Nelder-Mead Simplex algorithm, which is more robust to the initial point but converges very slowly. More information on advantages and limitations of different methods can be found in (SAS Institute Inc., (1995) SAS/IML Software: Changes and Enhancements through Release 6.11).

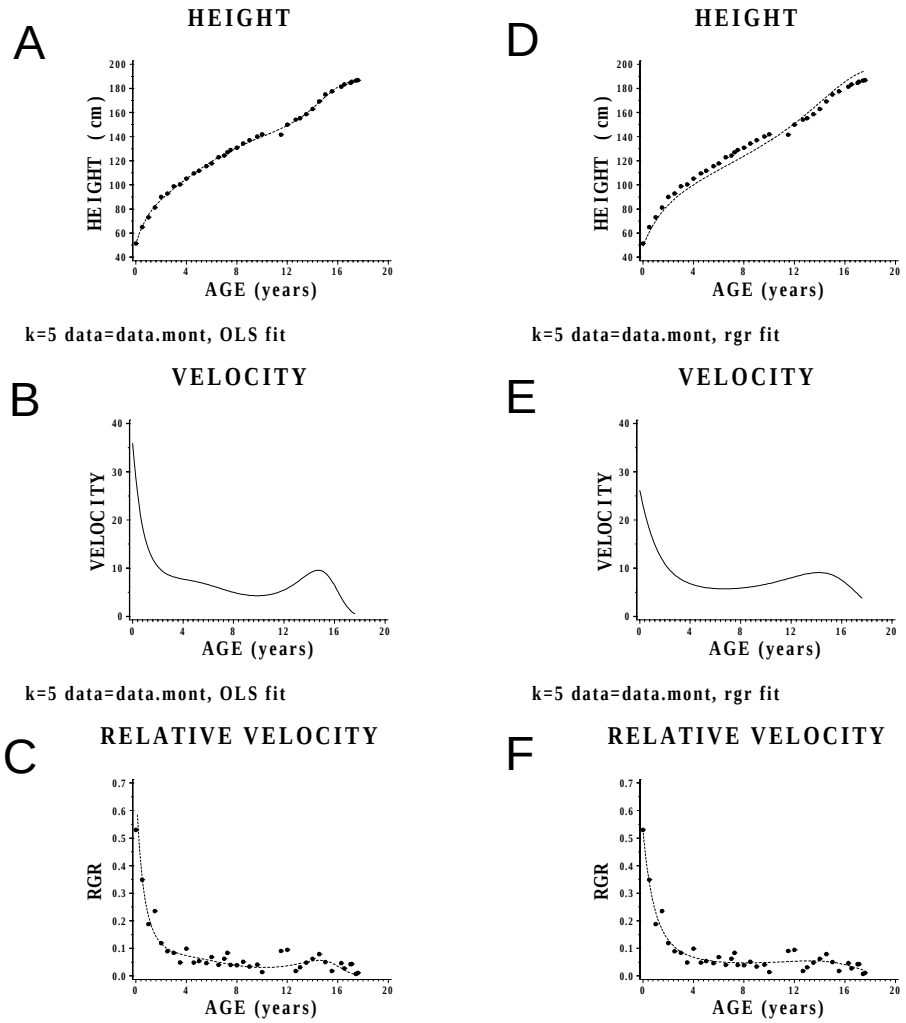

Figure1. Height, velocity and relative velocity plots for the data of the son of the Count de Montbelliard. Plots A-C were obtained by fitting the Polynomial Gompertz curve with K=5 to the height data. Plots D-F were obtained by fitting a fifth degree polynomial to the relative velocity data approximated from the height measurements.

The options offered for the non-linear programming functions in SAS/IML allow a great deal of control over their performance. For an advanced user it implies the possibility of tuning up the parameters which is important for an efficient solution in many of the optimization problems. At the same time, an inexperienced user who doesn't know much about the meanings of these options can easily use the default values without worrying about them. In this paper we give variouse examples of options usage.

All optimization routines in SAS/IML software have three groups of control parameters specified by three different vectors: *opt, tc* and *par.* These vectors can be passed as optional arguments to any function. As will many software packages, if any is omitted, the default value is assumed. The meanings of each component for these three vectors varies depending on the particular function it is used with.

In general, the *opt* vector controls general options like the amount of printed output for each iteration, particular Hessian update or line-search technique used. The *tc* vector determines the tolerance levels for different termination criteria used to stop optimization. The user is also allowed to define his own termination criterion as a separate module. The vector *par* specifies the internal parameters controlling each optimization method and the speed of its convergence. This is the most advanced option and requires some knowledge of the method's details. It allows a user to "tune-up" the algorithm to a particular problem and to improve its performance. More detailed descriptions for each of the components of *opt, tc* and *par* can be found in (SAS Institute Inc., (1995) SAS/IML Software: Changes and Enhancements through Release 6.11, p.171).

SAS IMPLEMENTATION OF THE PG GROWTH MODEL

Full-length code along with the description of the SAS macro %PGOLS which fits the *K*-th degree Polynomial Gompertz model to the growth data is available in (Perevozskaya I., Kuznetsova O.M. (2000)). Here, instead of describing the macro and its parameters, we outline the general structure of %PGOLS with emphasis on the optimization routine calls and concentrate on what was not highlighted in that paper — the computational

aspects.

In %PGOLS we used the Levenberg-Marquardt algorithm to obtain the ordinary least squares fit of the PG curve to the growth data. The convergence of that method can be very sensitive to the choice of a starting point. Even though one can set the maximal number of iterations to be a very large value to avoid getting an error message, it is essential to start with a good initial guess because each function evaluation is very costly (requires *n* calls to the numerical integration routine to evaluate $f(x_j)$, $j=1,\ldots,n.$) Therefore, starting from a poor initial guess may result in a very long computational time.

For the PG problem a good starting point can be obtained by using the RGR fit approach described in the introduction. The logarithm of the relative growth rate approximated from the data by using SAS/IML SPLINE and SPLINEV functions is fitted to the polynomial. This is a relatively inexpensive and simple procedure, which can be done using PROC NLIN. The resulting "crude" estimates of the polynomial coefficients are then improved using the computationally costly %PGOLS approach. In most examples, the convergence of the Levenberg-Marquardt routine was attained after 10-15 iterations.

Another important method to consider for improving convergence is scaling of the variables. This helps to reshape the objective function surface so that it is equally sensitive to the small changes in all variables. Formally, it is equivalent to making the components of the gradient vector approximately the same (of the same order). In practice, one should look at the gradient component estimates printed in the beginning of the algorithm and select the scaling constants to make the gradient components closer. An example of scaling is demonstrated in Tables 1-2. The difference between the gradient components for B3 and B0 is very big. After rescaling it was reduced which improved the speed of convergence considerably.

Parameter	Estimate	Gradient
b0	116.7804	0.02989
b1	-27.9360	0.37925
b2	2.16128	4.8673
b3	-0.55624	63.30215
A0	0.0001	1606

Table1. Parameters estimates (polynomial coefficients) and gradient for the Polynomial Gompertz model of order K=3 before scaling.

Parameter	Estimate	Gradient
b0	116.7804	0.02989
b1	-27.9360	0.37925
b2	21.6128	0.48699
b3	-55.6244	0.06329
A0	0.0001	1606

Table2. Parameters estimates (polynomial coefficients) and gradient for the Polynomial Gompertz model of order K=3 after scaling.

Example: The following statements can be used to call the Levenberg-Marquardt routine:

```
NOBS=nrow(resp);
optn={50, 5};
optn[1]=N_OBS;
/*the number of observations in the data
set*/
tc1={., ., ., 0, 0, 1e-3, 0, 0};
x0=beta_sc; /* starting point */
call nlplm(rc,xr,"object",x0)opt=optn tc=tc1;
beta_sc=xr;
/* beta_sc has OLS estimates now */
beta=beta_sc/scale;
/* return to unscaled parameters */
```

provided that earlier in the code one has defined the macro variable `&degree` and the SAS/IML variables : `resp` (a vector containing the response variable measurements), `age` (a vector of ages when measurements were made), `beta_sc` (a scaled vector of initial parameter estimates), and `"object"` (the least-squares function module). The first component of the option vector for the NLPLM call has to be the number of terms in the least square function (the number of observations in our case). Setting `opt`[2]=5 specifies the maximal amount of printout for each iteration. In this example the `tc1` vector is passed to the NLPLM function to select a specific termination criterion. Normally, optimization is finished when one of the several termination criteria is satisfied with the tolerance level given by the `tc` vector. Setting all but one values in `tc1` to zero results in ignoring the corresponding termination criteria, as in our case, where the only stopping criterion selected was max(gradient) <1e-03. The first three components of `tc1` vector are set to missing values to allow use of the default values.

SAS IMPLEMENTATION OF THE PROPORTIONAL ODDS MODEL

We defined the formal optimality criteria for the most efficient design as scalar functions of the Fisher's information matrix, which, in turn, depends on the unknown parameters, dose levels, design weights (proportions of patients allocated to each dose) and the number of different doses used in the study. The latter is usually not known in advance. A common practice is to find a design optimal for some "best guess" of the unknown parameters $\boldsymbol{\theta}=(\alpha_2, \alpha_3, \alpha_4, \beta)$. In other words, one has to minimize determinant or trace of the variance matrix $[C^T M^{-1}(d_1,\ldots,d_n;w_1,\ldots w_{n-1};\boldsymbol{\theta})C]$ with respect to the design $\xi=\{ d_1,\ldots,d_n;w_1,\ldots w_{n-1}\}$ and $\boldsymbol{\theta}$ being fixed at some value. This approach is often called "locally optimal design", since it relies on a single "best guess" of the parameter.

The following algorithm with a formal theoretical proof can be found in (Atkinson A.C.(1992) and Silvey S.D. (1980)). It computes the best design when the number of support points is unknown.

Step 0: Select some values of $\boldsymbol{\theta}=(\alpha_2, \alpha_3, \alpha_4, \beta)$ (the "best guess")

Step 1: Set n=2

Step 2: Using n dose levels find the best n-point design ξ_n^* by minimizing trace or determinant of the matrix $[C^T M^{-1}(d_1,\ldots,d_n;w_1,\ldots w_{n-1};\boldsymbol{\theta})C]$ subject to the constraints $0\le w_i \le 1$, $\sum w_i < 1$ $i=1,\ldots, n-1$.

Step 3: Plot the directional derivative function $d(M(\xi_n^*),x)$. This function can be interpreted as a gain in efficiency if the point x is added to the design ξ_n^*. Since the formula for $d(M(\xi_n^*),x)$ is very complex, we do not reproduce it here. The details and expressions of the directional derivatives for D_A-optimality and omnibus criteria can be found in (Perevozskaya I. (2001)).

Step 4: Verify the "global optimality" condition: if the candidate design ξ_n^* is indeed the most efficient design among all other possible designs then the values of $d(M(\xi_n^*),x)$ have to be non-

positive for every x and must equal zero at the points $x=d_1,\ldots,d_n$.
Step 5: If this condition is satisfied, then the algorithm stops and ξ_n^* is accepted as a solution. Otherwise, ξ_n^* is the best only among all n-point designs, but not truly optimal. In this case, a new dose level has to be added. Set $n=n+1$ and repeat Steps 2-5.

Full-length SAS code implementing a single pass of this algorithm would be too large for this paper. Instead, we just describe the main steps with some code examples.

Some of the model parameters are stored as macro variables and need to be defined in the very beginning:

```
%let criterion="omnibus";
/* select "Da-opt","Dopt" or "omnibus"
%let NPARMS=4;
%let NPTS=3; /*three-point design */
%let xmin=0;
/* plotting parameters for grid: */
%let xmax=20;
/* range and the number of grid points */
%let ngrid=200;
```

Others are stored as SAS/IML variables inside of the PROC IML:

```
proc iml;
reset noprint;
options ls=92;
PARMS={1 1 4 9};
/* PARMS vector should be        */
/*beta, alpha2,alpha3,alpha4 ....*/
design={0.02 2.26 4.64 0.3 0.3};
/* d1,d2,d3,w1,w2  */
initial=design;
gamma={0.6 0.4 0.3};
/*(gamma2,gmma3,gamma4)             */
/* determine the matrix C for Da-opt*/
/* and omnibus criteria             */
```

The following SAS/IML functions have to be defined before proceeding with optimization:

```
start I_point(x_point) global (PARMS);
/* information matrix evaluated at a single
point x_point */

start M_des(design)  global (PARMS);
/* full design information matrix */

start criteria(design) global (PARMS , C);
/* D,Da or omnibus criterion to */
/*be minimized               */

start predvar(point,design) global( PARMS,  C
);
/* directional derivative function used */
/*for graphical verification of         */
/*the optimality conditions              */
```

The boundary and linear constraints have to be specified by a matrix formed by adding the two upper rows containing the lower and upper boundaries for variables to the
matrix of linear constraints.

```
optn={0 3};/* minimization*/
%macro getconst;
  constr=J(2*&NPTS+1,3,.);
  lbound=J(&NPTS-1,1,0);
/* lower bound for weights */
  ubound=J(&NPTS-1,1,1);
/* upper bound for weights */
  ubound1=J(&NPTS,1,MTD);
/* upper bound for points */
  lincon=J(&NPTS+1,1,-1);
  lincon[&NPTS]=1;
  constr[(&NPTS+1):(2*&NPTS-1),1]=lbound;
/* for weights */
  constr[(&NPTS+1):(2*&NPTS-1),2]=ubound;
/* for weights */

  constr[(&NPTS+1):(2*&NPTS+1),3]=lincon;
  constr=t(constr);
%mend getconst;

%getconst;

des_ini=t(initial);
con=constr;
```

Here is an example of how the parameters of the Nelder-Mead simplex algorithm can be adjusted. Because of the possible slow convergence, the maximal number of iterations (tc[2]) has been increased to 4000. The final simplex radius tc[9] is used in the termination criteria for this algorithm. It is set to a very small value to improve the precision of the result.

```
/* set rhoend and max number of function
evaluations */
  tc=J(1,10,.); tc[9]=1e-10; tc[2]=4000;

/* setting up the GTOL criteria */
tc[4]=1e-10; tc[6]=1e-10;
par=j(1,10,.); par[2]=1; /* set rhobeg */
```

We apply the Nelder-Mead simplex algorithm (NLPMNS) and Quasi-Newton methods to the same objective function, with the same starting guess and control parameter vectors.
The resulting optimal solutions and return codes are stored in variables `optim1`, `rc1`, `optim2`, and `rc2`. These values are used by plotting procedures comparing results obtained by different methods.

```
call nlpqn(rc1,optim1,"criteria",des_ini,
optn,con)tc=tc par=par;
call nlpnms(rc2,optim2,"criteria",des_ini,
optn,con)tc=tc par=par;
```

The following calls to the plotting macros %IMLPLOT and %SASPLOT are required to create the plots of directional derivatives. %IMLPLOT macro creates SAS data sets from the IML vectors and %SASPLOT macro generates the graphic output from these data sets by calling PROC GPLOT.

```
%imlplot(forplot=forplot1,optim=optim1,
method=QuaziNewton,des_full=opt_des1,
dat_des=dat_des1,last=last1,suprem=suprem1 );
%imlplot(forplot=forplot2,optim=optim2,
method=Nelder-Mead,
des_full=opt_des2,dat_des=dat_des2,last=last,
suprem=suprem2);

%sasplot(forplot=forplot1,optim=optim1,
method=Quazi-Newton,des_full=opt_des1,
dat_des=dat_des1,last=last1,suprem=suprem1,
grname=genlocalQN);

%sasplot(forplot=forplot2,optim=optim2,
method=Nelder-Mead,des_full=opt_des2,
dat_des=dat_des2,last=last2,suprem=suprem2,gr
name=genlocalNM);
```

The parameters of both macros include the name of the data set to be plotted (`&forplot`), optimal design (`&optim`), optimization method used (`&method`), the data set containing values of the directional derivatives evaluated at each dose level (`&suprem`), and the name of a file containing the graph (`&grname`).

The complete SAS code implementing the algorithm for computing the local optimal design for the Proportional Odds model is available upon request.

CONCLUSION

SAS/IML software provides a powerful optimization package combined with matrix algebra tools and other numerical routines that can be used to solve many problems. Two examples considered demonstrated that the package works very well even for the computationally intensive problems. It is also very convenient to use because of the standardized syntax of all functions and the high degree of flexibility in numerical methods.

REFERENCES

Atkinson A.C., Donev A.N. (1992). Optimum Experimental Design. Oxford: Clarendon Press.

Babb J., Rogatko A., Zacks S (1998). Cancer phase I clinical trials: efficient dose escalation with overdose control. Statistics in Medicine 17:1103-1120

Perevozskaya I. (2001). Constrained Bayesian optimal design for phase I clinical trials. Doctoral dissertation, University of Maryland Graduate School, Baltimore

Perevozskaya I., Kuznetsova O.M. (2000). Modeling Longitudinal Growth Data and Growth Percentiles with Polynomial Gompertz Model in SAS® Software. SUGI25 Proceedings

SAS Institute Inc., (1995) SAS/IML® Software: Changes and Enhancements through Release 6.11, Cary, NC: SAS Institute Inc.

Silvey S.D. (1980). Optimal design. London: Chapman and Hall.

Zocci S.S., Atkinson A.C. (1999) Optimum experimental designs for multinomial logistic models. Biometrics 55: 437-444.

ACKNOWLEDGMENTS

The author would like to acknowledge Dr. Olga Kuznetsova for introducing her to the field of growth modeling , Professor Linda Haines for numerous suggestions and ideas she offered for the Proportional Odds Model, and Professor William F. Rosenberger for his endless patience and inspiration while guiding the author through her thesis research.

CONTACT INFORMATION

Your comments and questions are valued and encouraged.
For further information please contact: :

Dr. Inna Perevozskaya,
Department of Mathematics and Statistics
University of Maryland Baltimore County,
1000 Hilltop Circle, Baltimore, MD, 21250
Phone: (410)-455-2412
Fax: (410)-455-1066
Email: inna@math.umbc.edu